МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет електроніки та інформаційних технологій

Кафедра комп’ютерних наук

Кваліфікаційна робота магістра

ІНФОРМАЦІЙНО-АНАЛІТИЧНА ТЕХНОЛОГІЯ МОНІТОРИНГУ ВИРОБНИЦТВА ЕЛЕКТРОЕНЕРГІЇ

Здобувач освіти гр. ІН.м-13 Євгеній ПІВЕНЬ

Науковий керівник,
кандидат технічних наук, доцент,
доцент кафедри комп’ютерних наук Ігор ШЕЛЕХОВ

В.о. завідувача кафедри,
кандидат технічних наук, доцент,
доцент кафедри комп’ютерних наук Ігор ШЕЛЕХОВ

Суми 2022

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Факультет ЕлІТ Кафедра Комп'ютерних наук

Спеціальність 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ:

в.о. зав. кафедри _____

«_____»_____ 20_____р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Півню Євгенію Олеговичу

1. Тема роботи *Інформаційно-аналітична технологія моніторингу виробництва електроенергії* затверджую наказом по СумДУ від «____» _____ 20 ___ р. № _____
2. Термін здачі здобувачем кваліфікаційної роботи _____
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Вступ; 2) Інформаційний огляд; 3) Постановка завдання; 4) Вибір методу рішення; 5) Інформаційне та програмне забезпечення; 6)Висновок; 7) Література; 8) Додаток.*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р. Керівник _____

Завдання прийняв до виконання _____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання	Примітка
1	*Аналіз проблеми та постановка задачі*		
2	*Вибір методу рішення*		
3	*Інформаційне та програмне забезпечення*		
4	*Оформлення кваліфікаційної магістерської роботи*		

Здобувач вищої освіти _____

Науковий керівник _____

РЕФЕРАТ

Записка: 68 ст., 28рис., 2 табл., 1 додаток, 15 джерел.

Об’єкт дослідження — процес інформаційного аналізу і синтезу веб-орієнтованої системи моніторингу виробництва електроенергії.

Мета роботи – розробка комплексу методів, моделей та засобів проєктування веб-орієнтованої системи моніторингу виробництва електроенергії

Методи дослідження – методи проєктування інформаційних систем, методи нормалізації баз даних, об’єктно-орієнтована парадигма програмування, методи тестування інформаційних систем.

Результати – проведено розробку та програмну реалізацію веб-орієнтованої системи моніторингу виробництва електроенергії. При цьому виконано такі основні завдання: виконано інформаційно-аналітичний огляд сучасних сервісів моніторингу статистики електростанцій, розроблено інформаційну модель інформаційно-аналітичної технології моніторингу виробництва електроенергії, розроблено та нормалізовано структуру бази даних, обрано засоби програмної реалізації, розроблено і програмно реалізовано інформаційно-аналітичну технологію моніторингу виробництва електроенергії, перевірено працездатність розробленого веб-сервісу.

ВЕБ-СЕРВІС, MVC, APACHE, MSSQL, MYSQL, PHP, JAVASCRIPT, JQUERY, HTML, CSS, SMARTY

ЗМІСТ

ВСТУП

На сьогоднішній час навколо нас велике різноманіття сервісів, які складають велику частину нашого повсякдення. Від різноманітних мобільних додатків до ресурсів, які ми можемо використовувати ледь не щодня. Головна ціль даного виду розробок – максимально полегшити життя для будь-якого користувача в будь-якій сфері життя. Цінність веб-сервісів для бізнесу можна поділити на дві складові:

Внутрішня. Допомагає автоматизувати та спростити процес бізнесу. Наприклад, зробити звіт, налагодити обмін інформацією між відділами чи стежити за наявністю товару;

Зовнішня. Це взаємодія з користувачами. Наприклад, онлайн-запис на послуги, замовлення товарів чи послуг

Функціональність веб-програми та її застосування обмежується лише фантазією розробника. Але від того, як реалізований веб-сервіс і що він «уміє», багато в чому залежить успіх проекту. Тому потрібно підійти ретельно до кожного етапу розробки веб-сервісу, а також мати розрахунки по фінансовим витратам на розробку. На сучасному етапі веб-сервіс потребує розробки серверної частини, проектування бази даних та клієнтської частини. Кожен із даних етапів потребує значних часових та фінансових витрат, що необхідно враховувати перед початком роботи.

Інформаційно-аналітична технологія моніторингу виробництва електроенергії має в собі велику важливість, так як власник, вкладуючи гроші у виробництво електроенергії повинен бачити цілодобову статистику щодо своїх станцій. Дана розробка дозволить значно полегшити моніторинг параметрів електростанцій та отримувати якісний розрахунок витрат та прибутку від вкладень власника.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Загальний огляд веб-сервісів

Веб-сервіс - це програма в інтернеті, яка надає послугу або відповідає на певну вимогу користувача. Наприклад, електронна пошта надсилає листи, пошукова система Google шукає інформацію в інтернеті, а сайт з погодою показує прогноз. Сенс створення сайту – поділитися з аудиторією інформацією, показати себе та свою компанію[5].

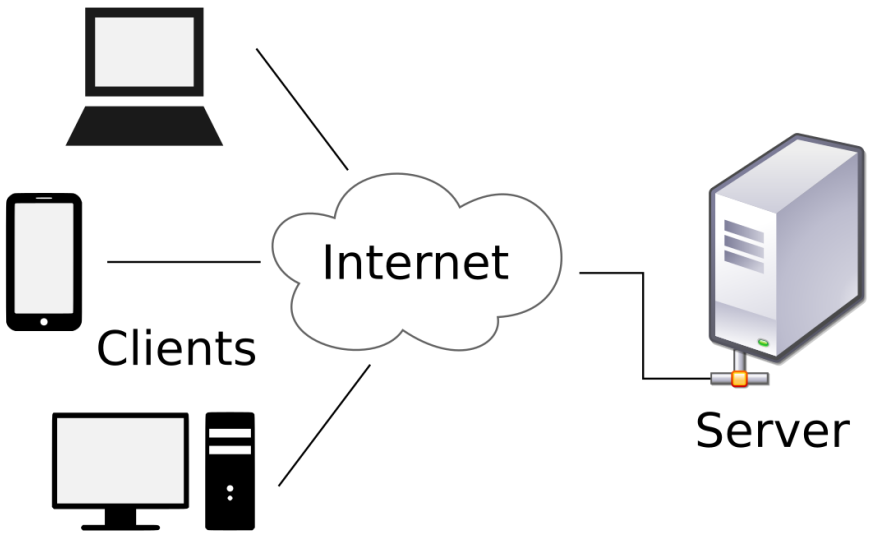

Рисунок 1.1 – Структура клієнт-серверної технології

Щоб отримати веб-сервіс, який буде відповідати всім вимогам на сьогоднішній день, краще віддавати перевагу індивідуальній розробці. Робота над додатком складається з кількох етапів:

1. Постановка конкретного завдання та збір даних. На цьому етапі аналізуємо цільову аудиторію, пропозиції конкурентів, визначаємо вимоги до сервісу.

2. Тестування бізнес-логіки за допомогою прототипу. Розробники підготують інтерактивні макети та продемонструють замовнику, як працюватиме програма.

3. Розробка дизайну. Робимо сервіс не лише привабливим, а й інтуїтивно зрозумілим для користувачів.

4. Програмування. Розробляємо гнучку архітектуру програми, продумуємо можливості масштабування проекту у майбутньому.

5. Внутрішнє випробування. «Обкатуємо» програму всілякими тестами, перевіряємо стійкість до високих навантажень.

6. Запуск. Після ретельної перевірки налаштуємо робочі сервери та запустимо ваш проект. Підключимо необхідні інструменти для аналітики та моніторингу.

Для більш детального розуміння веб-сервісів опишемо їх основні види та їхні переваги і недоліки. Існують три основні види веб-сервісів:

SPA - односторінкові програми. Інформація зосереджена на одній сторінці, а нові блоки динамічно підвантажуються. Завантажені дані будуть доступні офлайн.

MPA - багатосторінкові програми. Класичне рішення, за якого при зверненні до кожного розділу відкривається нова сторінка.

PWA – прогресивні програми. Відносно новий тип сервісів, що взаємодіє з користувачем як мобільний додаток. Ви можете винести його значок на робочий стіл та використовувати частину функцій офлайн.

Щоб визначитися з вибором типу програми для проекту, потрібно порівняти переваги та недоліки кожного з них[2].

Плюси SPA:

- Висока швидкість роботи;
- Можна реалізувати цікаву графіку, багатий та різноманітний інтерфейс;
- Простота кешування даних, робота офлайн.

Мінуси SPA:

- Складнощі з SEO: підняти сервіс у пошукових системах буде не так легко;
- Високе навантаження на браузер, при недостатній кількості оперативної пам'яті можуть виникати проблеми;
- Необхідність підтримки JavaScript на пристрої користувача: без неї програма просто не працюватиме;
- Вища вартість порівняно з MPA.

Плюси MPA:

- Звична для користувачів схема взаємодії: нова сторінка при кожному переході виглядає як звичайний сайт;
- Легка SEO-оптимізація: вам не знадобиться багато зусиль, як із SPA;
- Працює без підтримки JavaScript більш універсально;
- Немає обмежень щодо кількості інформації.

Мінуси MPA:

- Більш складна, довга розробка проти SPA;
- Менш гнучкий інтерфейс порівняно зі SPA.

Плюси PWA:

- Багатоплатформість: програма працює в будь-якій операційній системі;
- Висока швидкість роботи, можливість роботи офлайн, як, наприклад, у Google Docs;
- Швидка розробка можна переробити вже наявний сайт в PWA. Дешевше, ніж мобільні програми;
- Можливість встановити на комп'ютер та смартфон в один клік, отримувати Push-сповіщення. Займають мало пам'яті на пристрої, порівняно з мобільними програмами.

Мінуси PWA:

- Залежність від браузера: не всі браузери підтримують функції таких програм;
- Велика витрата батареї пристрою, на якому використовується PWA;
- Також можуть виникати складнощі із SEO[8].

1.2 Огляд подібних рішень

Головна мета розробки веб-сервісу – вирішити проблему користувача. Наприклад, за допомогою програми людина може:

- замовити їжу;
- викликати таксі;
- зробити грошовий переказ;
- купити квиток на літак;
- відстежити статус замовлення.

Заздалегідь варто подбати, щоб сервіс витримував великий наплив користувачів. Тому при створенні розробники вибудовують архітектуру так, щоб її легко було горизонтально масштабувати і розвивати.

В нашому випадку будемо розглядати подібні рішення моніторингу виробництва та споживання електроенергії для кращого розуміння можливих рішень та оцінки власного проекту[3].

Почнемо з проекту прогнозування виробництва електроенергії – Enercast.

Enercast надає точні прогнози виробництва електроенергії, які дозволяють клієнтам у всьому світі ефективно управляти своїми активами, забезпечувати стабільність мережі та підвищувати рівень свого бренду. Завдяки можливості тренувати прогнози для особливостей кожного окремого місця, даний веб-сервіс допомагає будувати децентралізовану енергетичну систему майбутнього.

Enercast є піонером у застосуванні штучного інтелекту для прогнозування виробництва відновлюваної енергії. Ключовим елементом у цьому є здатність ефективно обробляти великі обсяги даних про погоду. Дана система поєднує

численні моделі прогнозування погоди від міжнародних метеорологічних служб із даними вимірювань, що стосуються конкретного місця, щоб дізнатися про поведінку окремих рослин. На основі цих моделей додаток цілодобово надає надійні прогнози для конкретних заводів у всьому світі з резервних центрів обробки даних.

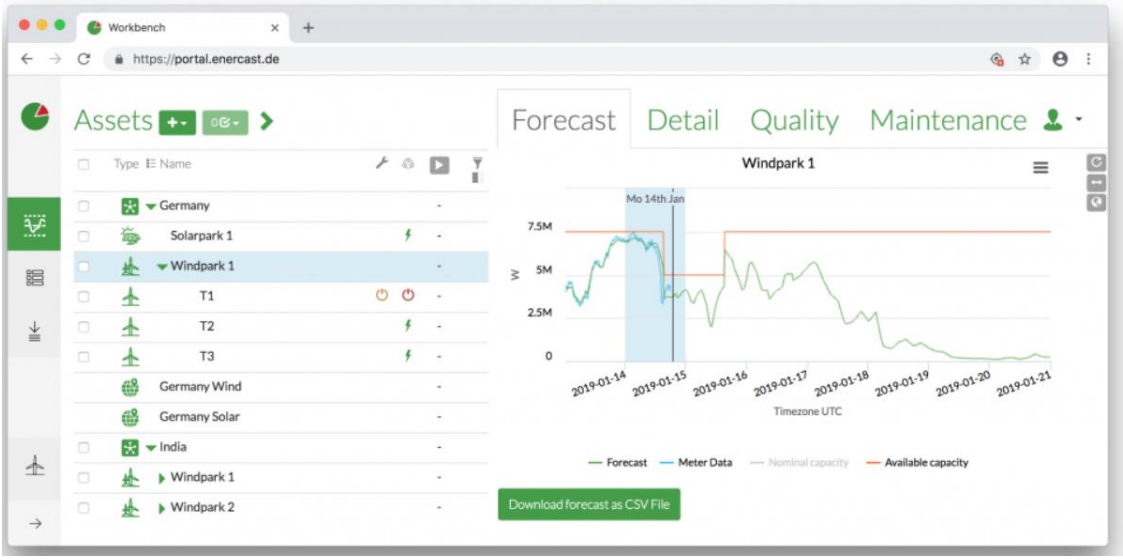

Рисунок 1.2 – Робота меню деталей плантації веб-сервісу Enercast

Використовуючи дану інтеграційну платформу, прогнози можна швидко та інтерактивно інтегрувати в бізнес-процеси клієнтів. Як результат відхилення та наслідки процесу, пов’язані з погодою, стають більш зрозумілими, що дозволяє оптимально керувати активами клієнтів. Дана відкрита інтеграційна платформа дозволяє об’єднувати різноманітну інформацію. Наприклад, якщо історичні дані про погоду, дані про споживання енергії та виробництво об’єднати з іншою якісною інформацією, штучні нейронні мережі зможуть знайти кореляції та закономірності, які дозволять скласти стабільний прогноз цін для внутрішньодобових, спотових і контрольованих енергетичних ринків.

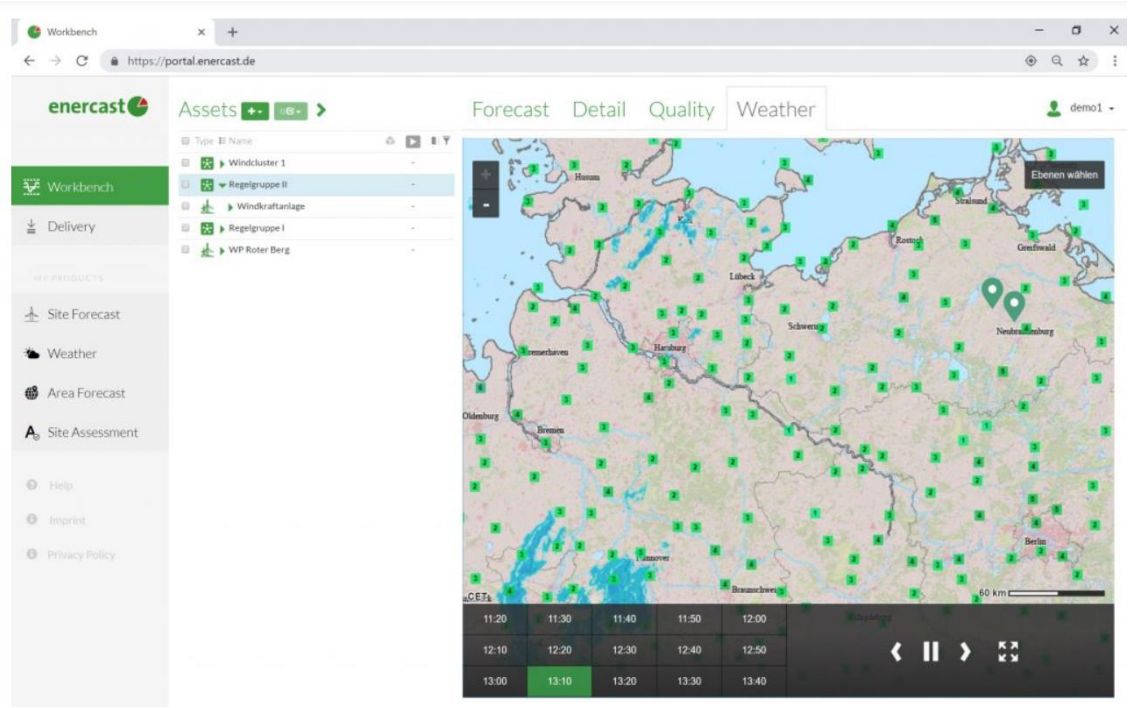

Рисунок 1.3 – Робота меню погоди плантації веб-сервісу Enercast

В проекті також існує API для інтерактивного зіставлення файлів. Уніфікована та стабільна якість даних досягається за допомогою інструменту очищення даних на основі штучного інтелекту.

Це дає змогу енергетичним і промисловим компаніям підвищити продуктивність і зменшити витрати на технічне обслуговування, поєднуючи й обробляючи широкий спектр джерел даних для створення нових ідей.

Використовуючи нашу відкриту платформу прогнозування, історичні дані про погоду та дані про поточне загальне енергоспоживання міста чи району, а також місцеві, але нестабільні обсяги споживання відновлюваної енергії можуть бути інтегровані, стандартизовані та корельовані.

Таким чином, можна краще передбачити структурно змінні навантаження на енергоспоживання, що, наприклад, може заощадити кошти для муніципальних комунальних підприємств і місцевих розподільників і підвищити стабільність мережі.

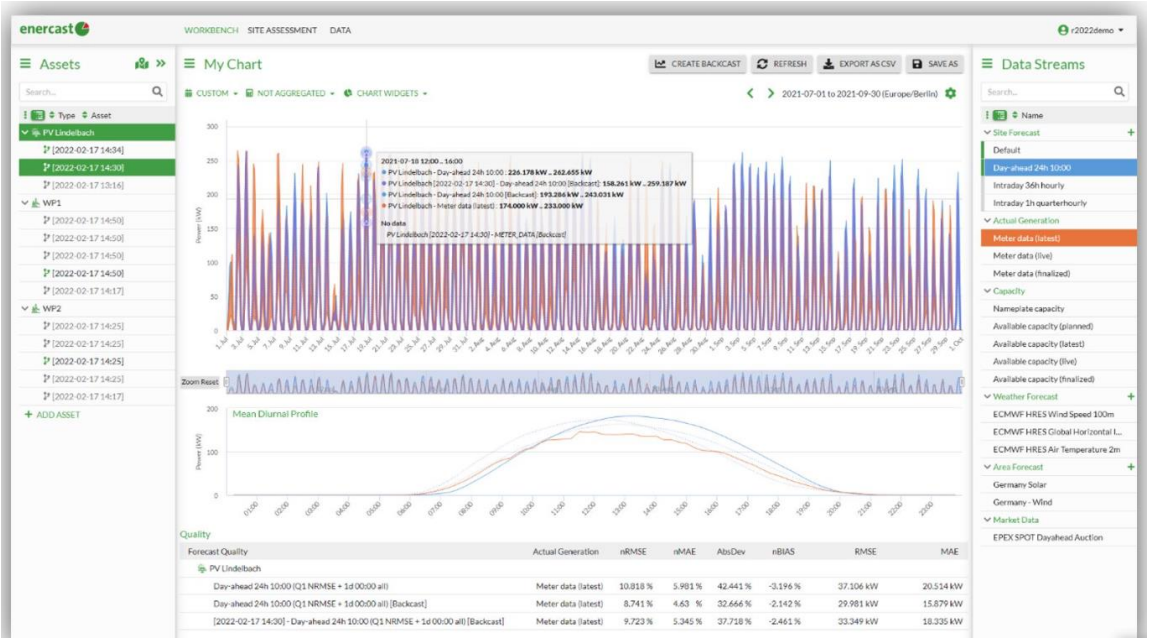

Рисунок 1.4 – Робота меню графіку плантації веб-сервісу Enercast

Enercast є провідним постачальником технологій для штучного інтелекту на основі погоди та цифрової трансформації відновлюваної енергії. Продукти SaaS із самонавчанням надають точні прогнози виробництва електроенергії для вітрових і сонячних електростанцій, що дозволяє інтегрувати відновлювану енергію в електромережі та енергетичні ринки по всьому світу.

Наступним на черзі розберемо платформу для інтеграції моніторингу Bold BI.

У сфері енергетики потрібно мати можливість відстежувати широкий спектр даних. З одного боку, потрібно контролювати енергоспоживання клієнтів. У той же час потрібно стежити за поточними даними виробництва електростанцій. Ці показники, звичайно, є вирішальними, оскільки вони гарантують, що ви можете задовольнити потреби клієнтів. Крім того, вони впливають на довгостроковий фінансовий успіх компанії. Однак відстеження такої кількості KPI одночасно може здатися непосильним завданням[10].

Незалежно від того, чи потрібно вам проаналізувати продуктивність турбіни чи оцінити рентабельність інвестицій для різних місць розташування ВЕС, Bold BI допоможе досягти даних цілей.

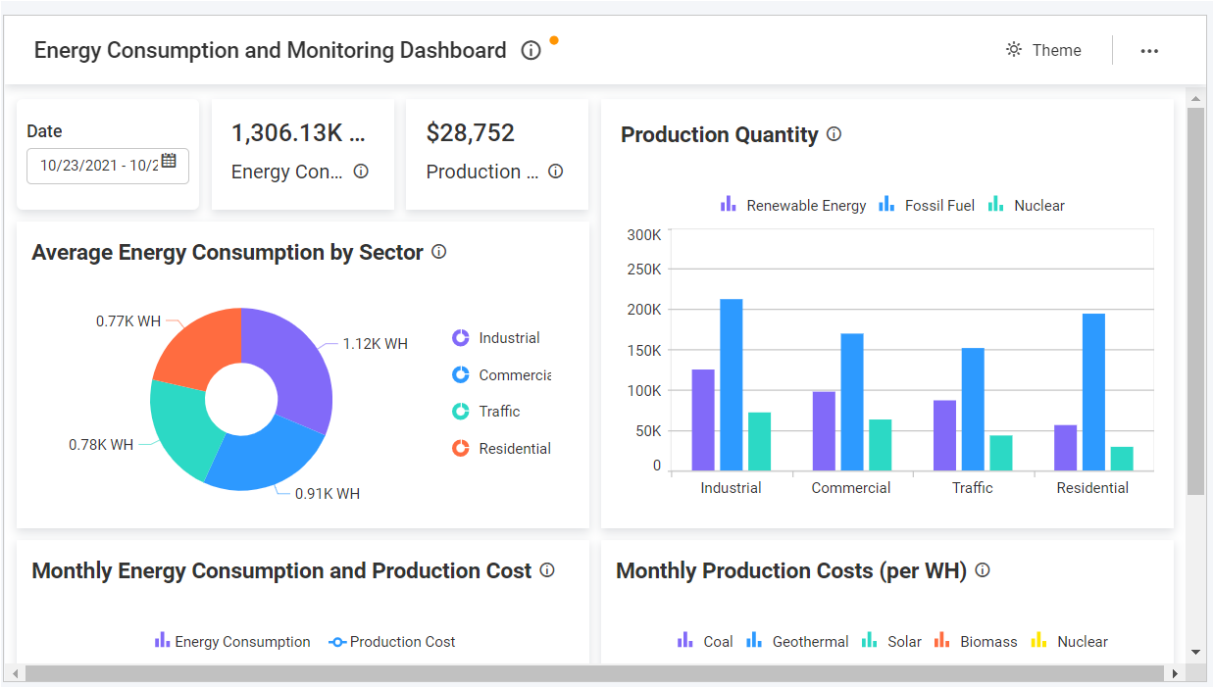

Рисунок 1.5 – Меню моніторингу споживання електроенегії проекту Bold BI

Інформаційна панель роботи електростанції Bold BI відображає поточні дані про виробництво плантацій електростанцій. Ця інформація має вирішальне значення, оскільки вона пропонує дані про виробництво заводів у реальному часі, дозволяючи глядачам миттєво визначати будь-які проблеми, які виникають. Ця інформаційна панель дозволяє клієнтам відповідати на запитання, які є ключовими для функціональності та фінансового стану електростанцій:

- скільки енергії виробляють електростанції? Чи відповідають вони потребам клієнтів?
- які заводи досягають своїх очікуваних показників виробництва, а які не досягають?
- чи ефективно працюють турбіни заводів?
- які заводи забезпечують найбільший прибуток від інвестицій? Як ми можемо покращити рослини, які є менш прибутковими?

Даний сервіс може збирати цю інформацію з кількох різних віджетів. Наприклад, можна легко побачити загальну потужність установок,

загальну вироблену енергію та їх цільову кількість. Потім ми можемо побачити середню частоту плантацій, температуру навколишнього середовища та вологість. Ці дані дають змогу зрозуміти, як функціонують всі системи. Крім того, є можливість аналізувати продуктивність кожної електростанції за допомогою віджета «Подробиці електростанції». Цей віджет показує розташування, тип турбіни, фактичний вихід порівняно з очікуваним, рівень виробництва та статус кожного заводу.

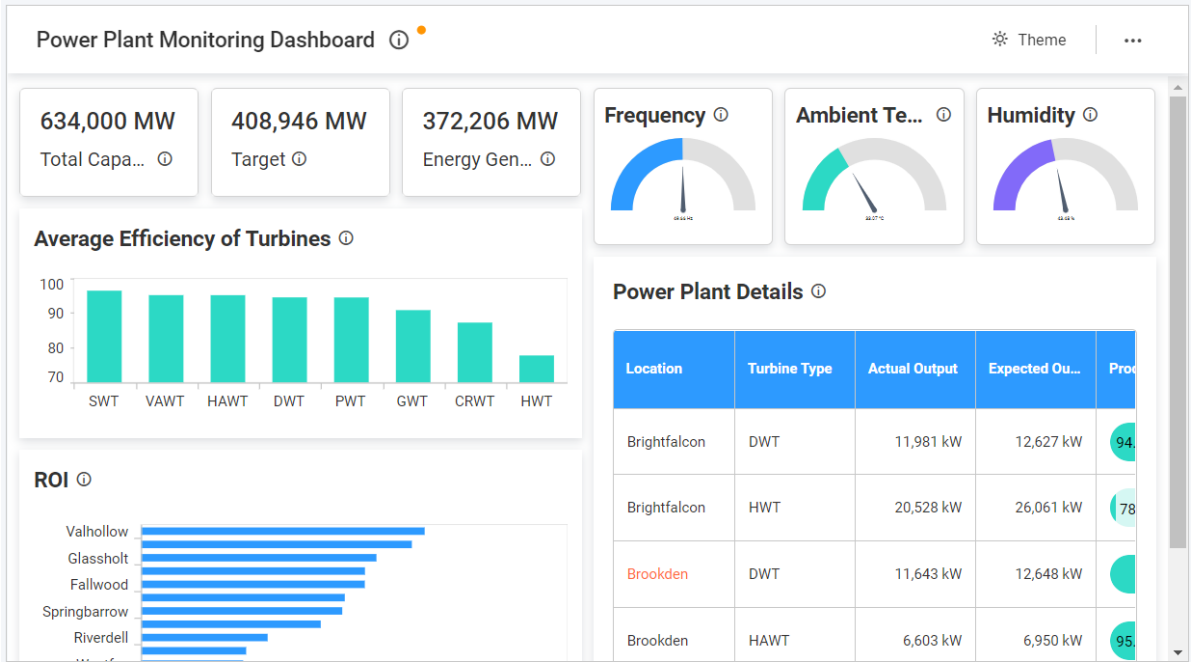

Рисунок 1.6 - Меню моніторингу виробництва плантацій проекту Bold BI

Віджет «Середня ефективність турбін» чітко описує продуктивність кожного типу турбіни. Це разом із віджетом «Подробиці електростанції» може допомогти глядачам проаналізувати ефективність електростанцій. Наприклад, якщо станція не досягає очікуваної потужності, ми можемо порівняти її тип турбіни з типом установки, яка відповідає очікуваній потужності, щоб визначити, чи існує кореляція. Таким чином, на цій інформаційній панелі відображаються ключові показники, які можуть допомогти керівникам аналізувати виробництво заводів у

режимі реального часу, забезпечуючи при цьому довгострокову ефективність і прибутковість.

1.3 Технології, які використовуються, та мови програмування

Розвиток Інтернет-сервісів постійно змінює спосіб розробки коду. Важливо використовувати правильну мову програмування, щоб швидко й найбільш доречно відносно вимог завершити проект.

Вибрана вами мова програмування значною мірою залежатиме від типу послуг, які ви збираєтеся надавати. Однак програми веб-сервісів змінюються в реальному часі, тому вам потрібно бути в курсі останніх тенденцій. Вибір правильної мови програмування для веб-служб дуже важливий, тому ви повинні бути дуже обережними. Під час веб-розробки можуть виникати деякі типові помилки[11].

Однією з найпоширеніших помилок є вибір мови, яка не відповідає конкретним потребам проекту. Це може призвести до уповільнення часу розробки та більш складних обхідних шляхів.

Щоб уникнути цієї проблеми, ви повинні ретельно оцінити потреби та варіанти використання для вашого проекту, перш ніж приймати остаточне рішення. Крім того, виберіть мову програмування, яка добре підтримується професіоналами галузі.

Для кращого розуміння мов програмування розберемо їх особливості та переваги і недоліки.

Python — це мова програмування загального призначення високого рівня, створена для бездоганної читабельності. Програмісти часто використовують його в наукових обчисленнях, аналізі даних, штучному інтелекті та машинному навчанні.

Переваги:

- Простий у використанні та вивченні: для початківців Python простий у використанні. Це мова програмування високого рівня, а її синтаксис схожий на англійську мову. Ці причини роблять мову легкою для вивчення та

адаптації. Порівняно з Java і C, у Python те саме завдання можна виконати за допомогою меншої кількості рядків коду. Завдяки легкому освоєнню принципи Python можна виконувати швидше порівняно з іншими мовами[9].

- Підвищення продуктивності: Python є дуже продуктивною мовою. Проста природа Python допомагає розробникам зосередитися на вирішенні проблем у ньому. Щоб зрозуміти синтаксис і поведінку мови програмування, користувачам не потрібно витрачати години, тому виконується більше роботи.
- Гнучкість: ця мова є дуже гнучкою, і тому вона дозволяє користувачеві пробувати нові інструменти. Користувачі можуть розробляти нові види програм, використовуючи мову програмування Python. Мова не обмежує користувача спробувати і наборі засобів для програмування. Інші мови програмування не надають такої гнучкості та свободи, тому Python є кращим у цих питаннях.
- Велика бібліотека: Python надає користувачеві величезну бібліотеку. Стандартна бібліотека Python величезна, і майже кожна функція, яку можна виконати, доступна в її бібліотеці. Це тому, що він має величезну підтримку спільноти та корпоративного спонсорства.
- Спільнота підтримки: Мова Python була створена багато років тому, і тому вона має зрілу спільноту, яка може підтримувати будь-який рівень розробки, починаючи від рівня початківців до рівня експертів. Для мови програмування Python доступно достатньо посібників, підручників та документації, які допомагають розробникам швидше та краще зрозуміти мову. Завдяки спільноті підтримки Python досить швидко розвивається порівняно з іншими мовами.

Недоліки:

- Швидкість: порівняно з Java або C швидкість Python нижча. Python — це інтерпретована мова, яка динамічно типізується. Для виконання коду кожен рядок коду має бути чітко впорядкований, оскільки мова інтерпретується. Це займає багато часу, а отже, уповільнює процес виконання. Динамічна структура Python також уповільнює його швидкість, тому що під час виконання коду

необхідно виконати зайву роботу. Тому у тих випадках, коли потрібне швидке прискорення, Python використовується не дуже часто.

- Споживання пам'яті: Python має дуже високе споживання пам'яті. Це тому, що він гнучкий до типів даних. Він використовує великий обсяг пам'яті. Python не є гарним вибором для завдань, де користувач хоче оптимізувати пам’ять, тобто це мова, яка потребує інтенсивного використання пам’яті.
- Розробка для мобільних пристроїв: Python сильний у серверних платформах і настільних комп’ютерах, а отже, це фантастична мова серверного програмування. Але це не підходить для мобільної розробки. Для мобільної розробки Python є крихкою мовою. Через це Python не має багато вбудованих програм для мобільних пристроїв, тому що він не економить пам’ять і має тривалу потужність для обробки. Carbonnelle — вбудована програма в Python.
- Доступ до бази даних: Python забезпечує просте програмування. Однак, коли він взаємодіє з базою даних, виникають деякі проблеми. У порівнянні з такими технологіями, як JDBC і ODBC, які є досить відомими, рівень доступу до бази даних мови програмування Python є примітивним і недостатньо розвиненим. Великі підприємства, яким зазвичай потрібна плавна взаємодія зі складними застарілими даними, не віддають перевагу використанню Python.
- Помилки виконання: користувачі Python згадували різні проблеми, з якими вони зіткнулися під час розробки мови. Оскільки мова Python динамічно типізована, тип даних змінної може бути змінений у будь-який час. Тому його потрібно тестувати частіше, а також є помилки в мові, яка відображається під час виконання.
- Простота: Python є простою та легкою у використанні мовою програмування, що також є недоліком мови. Користувачі Python настільки звикають до його легкого синтаксису та великої бібліотеки, що стикаються з проблемами під час вивчення інших мов програмування. Деякі користувачі також

вважають, що коди Java непотрібні через їх складність. Тому Python має дуже вразливий характер, і користувачі починають сприймати все легковажно[3].

PHP (Hypertext Preprocessor) — мова сценаріїв на стороні сервера, яка зосереджена на веб-розробці. Багато IT-фахівців використовують PHP для розробки та адміністрування веб-додатків різними способами[8].

Переваги:

- Відкритий і безкоштовний вихідний код: Однією з найважливіших переваг PHP є те, що він доступний для всіх. Люди можуть завантажити його з відкритого коду та отримати безкоштовно. Його можна завантажити будь-де та легко використовувати для розробки веб-додатків.
- Незалежність від платформи: ще одним важливим фактором є те, що оскільки програми на основі PHP можуть працювати на будь-якій ОС, як-от UNIX, Windows, Linux тощо, люди можуть використовувати їх, не турбуючись про платформу, на якій її можна використовувати.
- Легке завантаження : можна легко завантажити програми на основі PHP і підключити їх до бази даних. Люди в основному використовують його, оскільки він має високу швидкість завантаження, навіть якщо підключення до Інтернету повільне.
- Зручний для користувача: він має меншу криву навчання, і його можна швидко освоїти. Мова проста у використанні, і якщо хтось знає про програмування на C, він може швидко підхопити мову PHP для розробки програм.
- Стабільний: на відміну від інших мов сценаріїв, PHP дуже стабільний протягом багатьох років і має постійну підтримку. Також є можливість отримати допомогу щодо різних версій.
- Не потрібно багато коду: як ми згадували раніше, це проста мова, яку люди можуть використовувати для різних цілей. Він має таку якість, що його можна використовувати без необхідності писати довгі коди та складні структури для будь-якої події веб-додатку.

- Гнучкий: він дуже гнучкий, і люди можуть легко використовувати його, щоб поєднувати його функції з різними іншими мовами програмування. Вони можуть використовувати програмні пакети як найефективнішу технологію для кожної функції.
- Збільшення можливостей працевлаштування: Оскільки PHP дуже популярний, з'явилося багато розробників і спільнот розробників, які знають цю мову. Люди, які знають просту мову, можуть стати потенційними кандидатами на роботу.
- З'єднання з базою даних: він має вбудоване з'єднання з базою даних, яке допомагає з'єднувати бази даних і взагалі зменшити час і витрати на розробку веб-додатків або сайтів.
- Підтримка бібліотеки: PHP має потужну підтримку бібліотеку, за допомогою якої можна використовувати різні функціональні модулі для представлення даних[12].

Недоліки:

- Питання безпеки: оскільки PHP є відкритим кодом, він не є безпечним. Текстовий файл ASCII легко доступний, і кожен може знайти його.
- Не підходить для розробки гігантських веб-додатків: якщо хтось хоче розробити гігантський веб-додаток на основі даних, це неможливо зробити за допомогою PHP. Їм доведеться б використовувати інші мови програмування.
- Слабкий: PHP слабкий і іноді може спричиняти помилки. Це може призвести до того, що користувачам будуть доступні неправильні дані.
- Додаткове навчання: щоб використовувати вбудовані функції PHP, необхідно знати структуру PHP. Якщо хтось про це не знає, він буде змушений написати додаткові коди.
- Не дозволяє змінювати або модифікувати: неможливо модифікувати або змінити основну поведінку онлайн-додатків, оскільки PHP це не дозволяє.

- Поганий фреймворк: фреймворки PHP не еквівалентні за поведінкою порівняно з іншими такими мовами. Отже, його продуктивність і функції можуть страждати порівняно з іншими мовами програмування.
- Низька продуктивність: PHP не підтримує використання багатьох функцій одночасно. Використання додаткових функцій або інструментів PHP може призвести до низької продуктивності під час розробки онлайн-програм.
- Наявність простіших мов програмування : хоча PHP є потужною мовою програмування, існує багато інших мов, які підтримуються великою спільнотою та довідковою документацією, з якими легше працювати для веб-додатків[14].

JavaScript — це динамічна мова програмування, яка використовується для веб-розробки, у веб-додатках, для розробки ігор тощо. Це дозволяє реалізувати динамічні функції на веб-сторінках, які неможливо зробити лише за допомогою HTML і CSS.

Багато браузерів використовують JavaScript як мову сценаріїв для виконання динамічних завдань у веб-сторінці. Щоразу, коли ви бачите спадаюче меню, що відкривається, додатковий вміст, доданий на сторінку, і динамічну зміну кольорів елементів на сторінці, якщо назвати декілька функцій, ви бачите ефекти JavaScript.

Переваги:

- Простий − JavaScript простий для розуміння та сприйняття. І користувачі, і розробники зрозуміють структуру простою. Крім того, його дуже легко реалізувати, заощаджуючи веб-розробникам масу грошей під час створення динамічного вмісту.
- Швидкість − це «інтерпретована» мова, вона скорочує час, необхідний для компіляції іншими мовами програмування, такими як Java. Ще один сценарій на стороні клієнта — JavaScript, який прискорює виконання програми, усуваючи час очікування підключення до сервера. Незалежно від того, де розміщено JavaScript, він завжди виконується в клієнтському середовищі, щоб зменшити використання пропускної здатності та прискорити виконання.

- Інтероперабельність – Оскільки JavaScript легко інтегрується з іншими мовами програмування, багато розробників вважають за краще використовувати його для створення різноманітних програм. Його може містити будь-яка веб-сторінка або скрипт іншої мови програмування.
- Завантаження сервера – Перевірку даних можна виконати в самому браузері, а не пересилати їх на сервер, оскільки JavaScript працює на стороні клієнта. Весь веб-сайт не потрібно перезавантажувати у разі будь-яких розбіжностей. Браузер оновлює лише вибрану область сторінки.

Недоліки:

- Не вдається дебажити – хоча деякі редактори HTML дозволяють дебажити, вони не такі ефективні, як редактори для C або C++. Крім того, розробнику важко з’ясувати проблему, оскільки браузер не відображає жодних помилок.
- Неочікувана зупинка візуалізації – весь код JavaScript веб-сайту може припинити рендеринг через єдину помилку в коді, хоча користувачеві здається, що JavaScript помилка відсутня. Однак браузери часто допускають ці помилки.
- Безпека на стороні клієнта − користувач може бачити код JavaScript; нею можуть зловживати інші користувачі. Ці дії можуть передбачати анонімне використання вихідного коду. Крім того, на веб-сайт дуже просто вставити код, який погіршує безпеку даних, що передаються через веб-сайт.
- Успадкування − JavaScript не підтримує множинне успадкування; підтримується лише одне успадкування. Ця властивість об'єктно-орієнтованих мов може бути необхідною для деяких програм.
- Підтримка браузера – залежно від браузера JavaScript інтерпретується по-різному. Тому перед публікацією код потрібно запустити на різних платформах. Розробнику також потрібно перевірити старі браузери, оскільки вони не підтримують деякі нові функції.

Вибір мови програмування для вашого веб-сайту або програми – це те, що клієнт повинен обговорити з вашою компанією, що спеціалізується на веб-розробці. Досвідчені розробники зможуть запропонувати найкращі варіанти для досягнення ваших конкретних цілей. Не існує такої речі, як «найновіша» мова програмування для послуг веб-дизайну та розробки. Кожен доступний сьогодні варіант, навіть ті, що тут не перераховані, має багато плюсів і деякі мінуси. Вибір мови має ґрунтуватися на масштабах, бюджеті та меті вашого проекту[9].

1.4 Постановка задачі

Мета кваліфікаційної роботи магістра – розробка методів та засобів інформаційно-аналітичної технології моніторингу виробництва електроенергії, що дозволять спроектувати і реалізувати інформаційно-аналітичну технологію моніторингу виробництва електроенергії. При цьому основними завданнями роботи будуть:

1) Розробка інформаційної моделі функціонування веб-сервісу статистики виробництва електроенергії
2) Розробка структури бази даних
3) Визначення бізнес-логіки
4) Вибір засобів програмної реалізації
5) Програмна реалізація інформаційно-аналітичної технології моніторингу виробництва електроенергії
6) Тестування системи

2 ВИБІР МЕТОДІВ РІШЕННЯ ЗАДАЧІ

2.1 Інформаційна модель

Для розуміння предмету проекту більш детально представимо його у вигляді інформаційної моделі. Створення моделі відбулось за допомогою програми Paint. На рисунку 2.1 зображено представлення UML діаграми зв'язку користувачів: адміністратор, зареєстрований користувач, незареєстрований користувач.

Рисунок 2.1 – Діаграма UML зв'язку користувачів

На даній схемі представлений повний набір користувачів, які є доступними в проекті. Як можемо бачити, авторизований користувач має лише можливість перегляду турбін. Неавторизований користувач не має жодних прав, так як проект розрахований лише на невеликий перелік осіб, які мають доступ до даного проекту. На схемі вказаний користувач з правами admin, що має повний набір прав в проекті. Для нього є можливість переглядати турбін, редагувати їх, реєструвати користувачів в системі, переглядати їх та редагувати дані турбін в системі.

2.2 Розробка бази даних

На основі початкових умов в процесі розроблення проекту була створена база даних. Створення відбувалось за допомогою спеціальних SQL скриптів (forumizer), які запускались з терміналу як окремі SQL файли.

Згідно вимог, що були поставлені для необхідного функціонування статистики виробництва електроенергії була створена наступна таблиця 2.1:

Таблиця 2.1 – Опис таблиць бази даних

Таблиця	**Призначення**
adminloginlogout	Зберігає дані про авторизацію користувачів з правами admin
admins	Зберігає дані про користувачів з правами admin
brands_fields	Зберігає назви полів певного бренду турбін
brand_field_groups	Зберігає групи полів певного бренду турбін
cache	Зберігає дані кешу певних турбін
plants	Зберігає дані плантацій
loginlogout	Зберігає дані про авторизацію користувачів
brands	Зберігає дані брендів користувачів
turbines	Зберігає дані турбін
db_version	Зберігає дані версії бази даних
tokens	Зберігає токени користувачів
users	Зберігає дані користувачів

Під час розроблення проекту була створена ER діаграма, яка дозволяє описати концептуальну схему за допомогою конструкцій блоків. На рисунку 2.2 розташована ER модель, що містись в собі декілька найголовніших частин проекту.

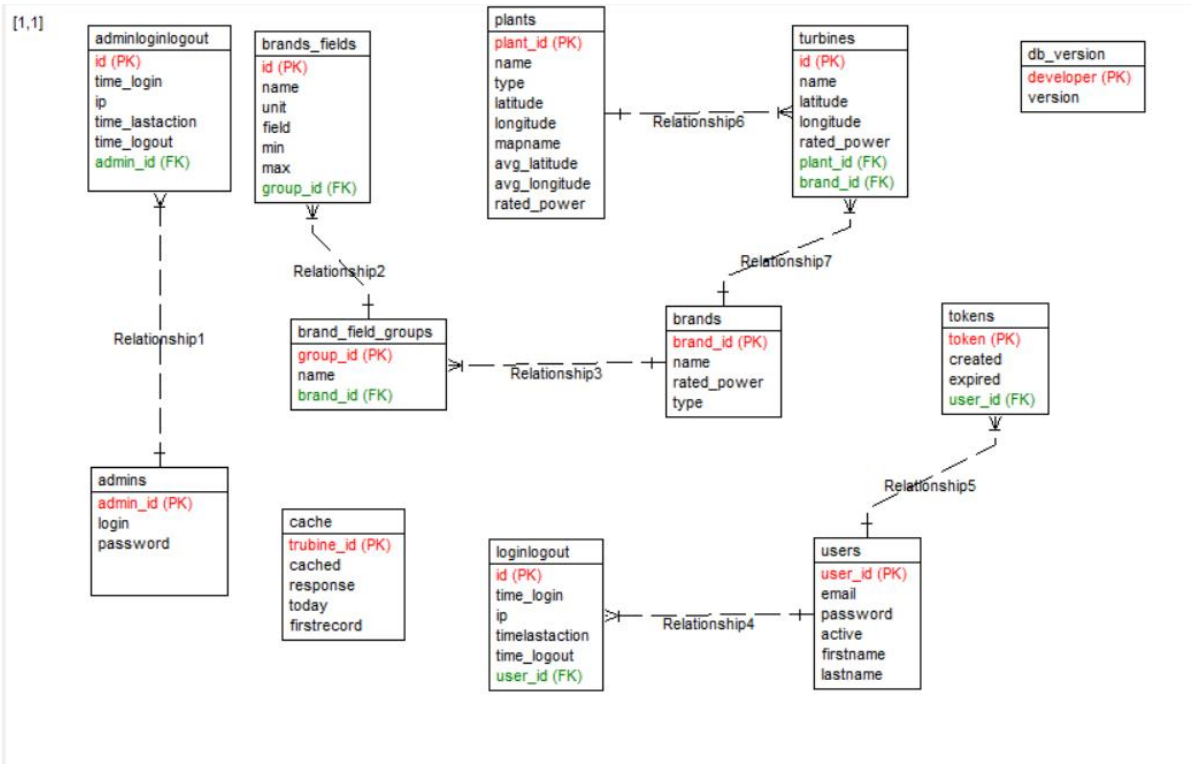

Рисунок 2.2 - ER модель статистики електростанцій

Також додатково до бази MySQL, яка зберігає статичні дані, в системі також існує база даних MSSQL, що містить в собі динамічні дані турбін у вигляді окремих таблиць. Ця база даних заповнюється окремо адміністраторами, дата останнього оновлення якого зберігається в окрему колонку таблиць [RealDate]. Для кращого її розуміння опишемо загальну ER схему всіх таблиць, що містяться в даній базі на рисунку 2.3.

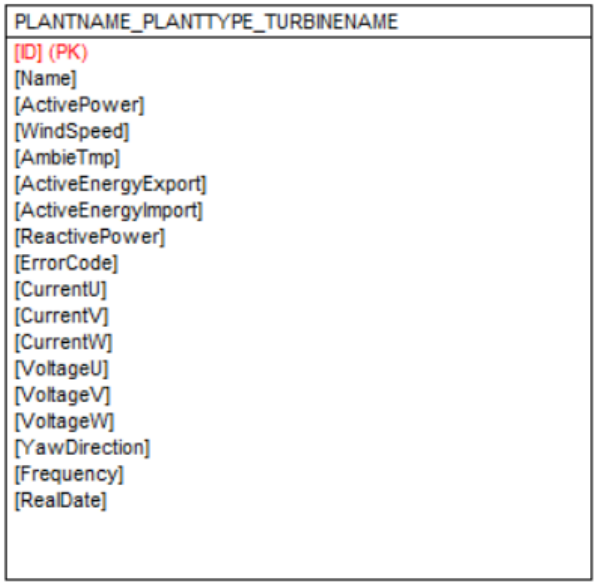

Рисунок 2.3 - ER модель представлення таблиць динамічних даних

На рисунку 2.3 зображена загальна схема всіх таблиць бази даних MSSQL. З назви таблиці PLANTNAME_PLANTTYPE_TURBINENAME випливає, що кожна таблиця в даній базі складається з трьох приставок. PLANTNAME відповідає за назву плантації, PLANTTYPE відповідає за тип плантації (вітряні станції, або геотермальні) та TURBINENAME – безпосередньо назва турбіни, до якої відносятся дані. Для прикладу, можемо обрати таблицю, яка використовується в режимі реального часу - AIRRES_W_T01. В даній таблиці AIRRES назва плантації, W – перша буква типу плантації (WIND) та T01 – назва турбіни.

Далі опишемо в таблиці 2.2 атрибути загальної таблиці для кращого розуміння проекту.

Таблиця 2.2 – Опис атрибутів бази даних

Атрибути	**Призначення**
[Id]	id відповідної турбіни
[Name]	Назва турбіни
[ActivePower]	Активна потужність виробництва
[WindSpeed]	Швидкість вітру
[AmbieTmp]	Температура турбіни
[ActiveEnergyExport]	Активна потужність експорту
[ActiveEnergyImport]	Активна потужність імпортування
[ReactivePower]	Реактивна потужність виробництва
[ErrorCode]	Код помилки
[CurrentU]	Струм відділу U
[CurrentV]	Струм відділу V
[CurrentW]	Струм відділу W
[VoltageU]	Вольтаж відділу U
[VoltageV]	Вольтаж відділу V
[VoltageW]	Вольтаж відділу W
[YawDirection]	Напрямок вітру
[Frequency]	Частота
[RealDate]	Поточна дата даних

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Вибір мов програмування

В розробленому сервісі для розроблення проекту на основі поданої інформаційної моделі використовуються дві основні мови програмування, це PHP та JavaScript. В пункті 1.3 було наведено їх особливості недоліки та переваги. В цьому пункті буде більш детально описано їх можливості та особливості, що стануть в нагоді під час розробки даного проекту.

PHP. Це серверна сценарна мова з відкритим кодом, яку багато розробників використовують для веб-розробки. Це також мова загального призначення, яку можна використовувати для створення багатьох проектів, у тому числі графічних інтерфейсів користувача. Абревіатура PHP спочатку означала Personal Homepage. Але тепер це рекурсивна абревіатура Hypertext Preprocessor. Перша версія PHP була запущена 26 років тому. Зараз це версія 8, випущена в листопаді 2020 року, але версія 7 залишається найпоширенішою[4].

PHP працює на двигуні Zend, який є найпопулярнішою його реалізацією. Існують також деякі інші реалізації, наприклад parrot, HPVM (Hip Hop Virtual Machine) і Hip Hop, створені компанією Facebook. PHP в основному використовується для створення веб-сервісів. Він працює в браузері, а також може працювати в командному рядку. Отже, якщо вам не хочеться показувати вихідний код у браузері, ви можете показати його в терміналі. Багато відомих компаній і технологічних гігантів використовують PHP для запуску своїх серверів і створення багатьох неймовірних речей[15].

- Facebook: Facebook використовує PHP для роботи свого сайту. У свою чергу, компанія зробила внесок у спільноту, створивши реалізацію, відому як Hip Hop для PHP.
- Вікіпедія: одне з найбільших у світі джерел інформації на будь-яку тему, Вікіпедія побудована на PHP.

- Системи керування вмістом (CMS): найпопулярніша у світі система керування вмістом, WordPress, побудована на PHP. Інші системи керування контентом, такі як Drupal, Joomla та Magento, також побудовані на PHP. Shopify також працює на PHP.
- Платформи веб-хостингу: багато платформ веб-хостингу, такі як BlueHost, Site ground і Whogohost, запускають свої сервери хостингу за допомогою PHP.

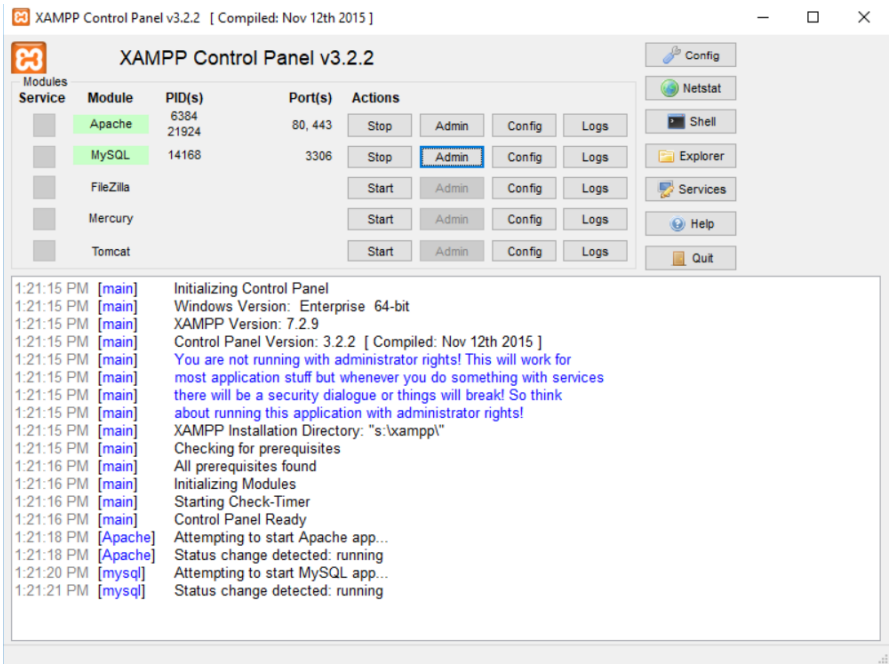

Рисунок 3.1 – Панель контролю серверу XAMPP (Apache)

PHP залишається актуальною та широко використовуваною мовою веб-розробки. Незважаючи на побоювання та дискусії про те, чи він все ще має вагу як мова програмування, розробники PHP продовжують непогано заробляти на роботі з цією мовою. Отже, здається, що PHP нікуди не подінеться найближчим часом[1].

Javascript. Це динамічна мова комп'ютерного програмування. Він легкий і найчастіше використовується як частина веб-сторінок, реалізації яких дозволяють сценарію на стороні клієнта взаємодіяти з користувачем і створювати динамічні сторінки.

Це інтерпретована мова програмування з об’єктно-орієнтованими можливостями. JavaScript працює на стороні клієнта в Інтернеті, який можна використовувати для проектування/програмування поведінки веб-сторінок у разі виникнення події. JavaScript — це проста у вивченні, а також потужна мова сценаріїв, яка широко використовується для керування поведінкою веб-сторінок.

Всупереч поширеній помилковій думці, JavaScript не є «інтерпретованою Java». У двох словах, JavaScript — це динамічна мова сценаріїв, яка підтримує створення об’єктів на основі прототипу. Основний синтаксис навмисно подібний до Java і C++, щоб зменшити кількість нових концепцій, необхідних для вивчення мови. Мовні конструкції, такі як оператори if, цикли for і while, а також блоки switch і try ... catch функціонують так само, як і в цих мовах (або майже).

JavaScript може функціонувати і як процедурна, і як об’єктно-орієнтована мова. Об’єкти створюються програмно в JavaScript шляхом додавання методів і властивостей до порожніх об’єктів під час виконання, на відміну від синтаксичних визначень класів, поширених у скомпільованих мовах, таких як C++ і Java. Після створення об’єкта його можна використовувати як план (або прототип) для створення подібних об’єктів.

Динамічні можливості JavaScript включають конструкцію об’єктів під час виконання, списки змінних параметрів, функціональні змінні, динамічне створення сценаріїв (через eval), самоаналіз об’єктів (через for ... in) і відновлення вихідного коду (програми на JavaScript можуть декомпілювати тіла функцій назад у вихідний текст)[11].

Функціонал проєкту реалізований на сервері та android додатку. На сервері знаходиться веб-сайт, доступний за певною адресою. Android додаток зв’язаний з API, що реалізований на серверній частині. Архітектура проекту реалізована за популярною моделлю MVC (модель-відображення-контроллер). Дана архітектура надає модульності при організації коду і нарощуванні функіоналу та інтуїтивності при розробці проєкту.

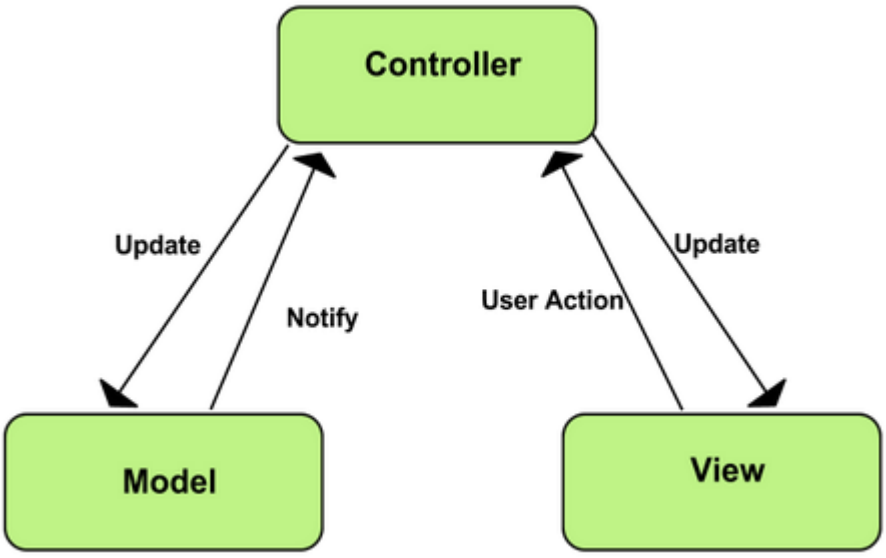

Рисунок 3.2 – Структура моделі MVC

Під час реаліазації проєкту була використана мова PHP у взаємодії з мовою JavaScript без використання будь-яких фреймворків. В даному випадку дана взаємодія мов програмування показує найкращий результат при реалізації веб-проєктів такого типу. Для PHP був використаний допоможній плагін Smarty для роботи з представленнями у схемі MVC[6].

Smarty — це механізм шаблонів для PHP, який полегшує відокремлення презентації (HTML/CSS) від логіки програми. Це означає, що PHP-код є логікою програми та відокремлений від презентації.

Під капотом Smarty компілює копії шаблонів як скрипти PHP. Таким чином ви отримуєте переваги синтаксису тегів шаблону та швидкості PHP. Компіляція відбувається один раз під час першого виклику кожного шаблону, а потім скомпільовані версії використовуються з цього моменту.

Smarty подбає про все за вас, дизайнер шаблонів лише редагує шаблони Smarty і ніколи не потребує керування скомпільованими версіями. Завдяки такому підходу шаблони зручні в обслуговуванні, а час виконання надзвичайно швидкий. Оскільки скомпільовані версії є PHP, прискорювачі операційного коду,

такі як APC або ZendCache, продовжують працювати над скомпільованими сценаріями.

Дизайн Smarty в основному керувався такими цілями:

- чітке відокремлення презентації від коду програми;
- PHP бекенд, інтерфейс шаблону Smarty;
- доповнюють PHP, а не замінюють його;
- швидка розробка/розгортання для програмістів і дизайнерів;
- швидкий і простий в обслуговуванні;
- синтаксис простий для розуміння, знання PHP не потрібні;
- гнучкість для індивідуальної розробки;
- безпека: ізоляція разом з PHP;
- безкоштовний, з відкритим кодом.

Сервер веб-сервісу написаний на мові програмування PHP. Для динамічного оновлення даних в проекті використовується мова програмування JavaScript у поєднання з базою даних MSSQL.

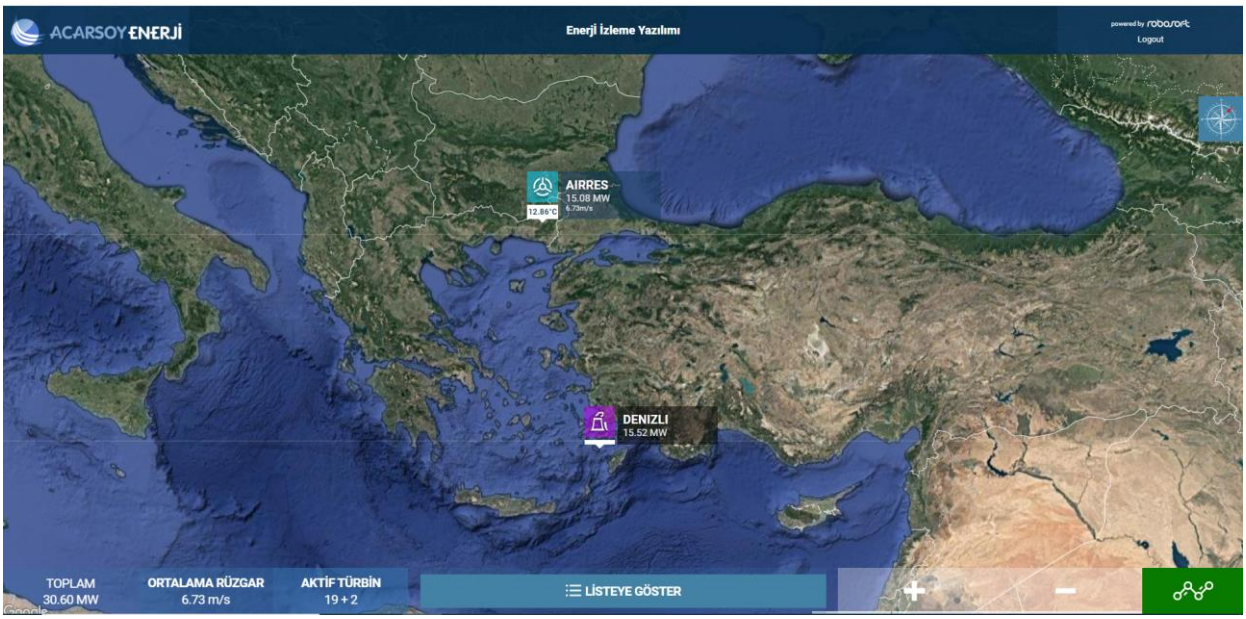

Рисунок 3.3 – Головна сторінка проекту

Для статичних даних, таких як наприклад певних даних турбін, а також назв параметрів використовується база даних MySQL. Загалом для проекту такого типу, з невеликим навантаженням таке поєднання є ідеальним.

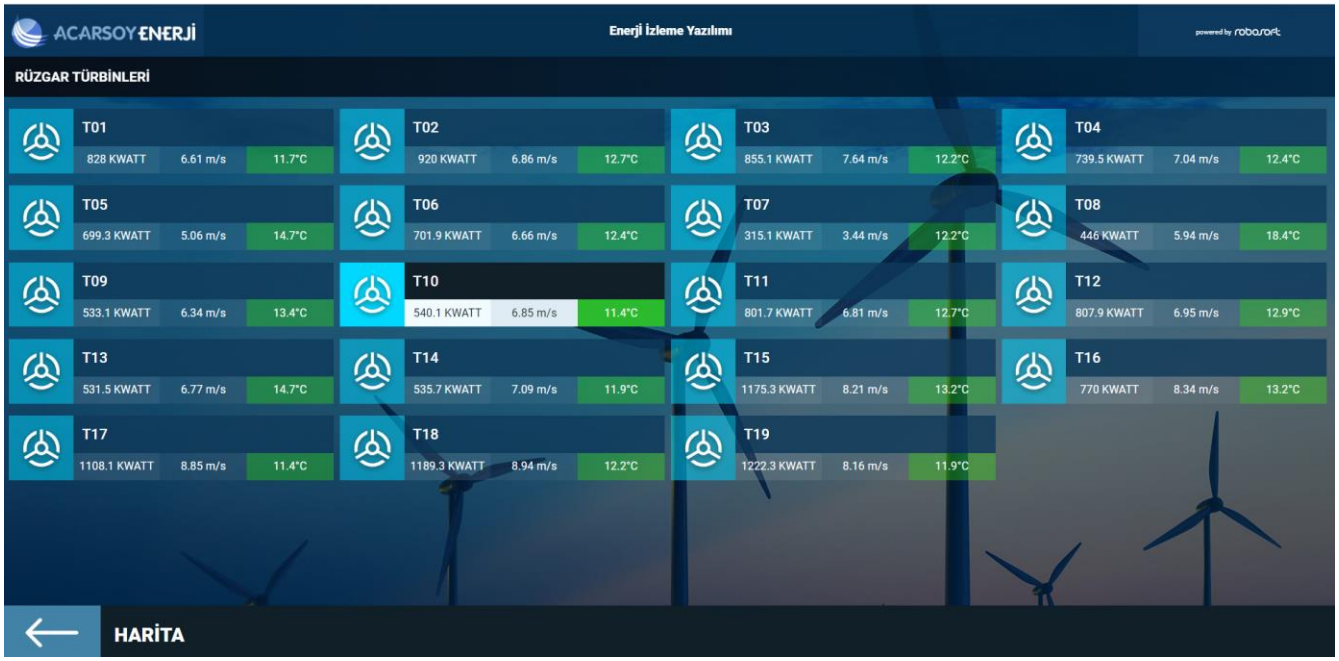

Рисунок 3.4 – Сторінка списку турбін

Проект розрахований лише на невеликий перелік осіб, що зацікавлені в інформації про стан свого бізнесу, тому головна задача функціональність, це швидкість, інтуїтивність інтерфейсу та модульність.

Архітектура даного проекту побудована саме таким чином, щоб мати можливість зручно нарощувати функціонал, та мати гарну швидкодію при постійному оновленні даних[7].

Кастомна архітектура, написана на мові PHP надає зручну функціональність, яка дає можливості якісно нарощувати модульність проекту, і одночасно не мати втрат у швидкодії. Для вдалого поєднання JavaScript та PHP використовуються AJAX запити, що відправляються від скриптів до окремих PHP файлів, що несуть в собі код, який має на меті збирати динамічні дані з бази даних MSSQL.

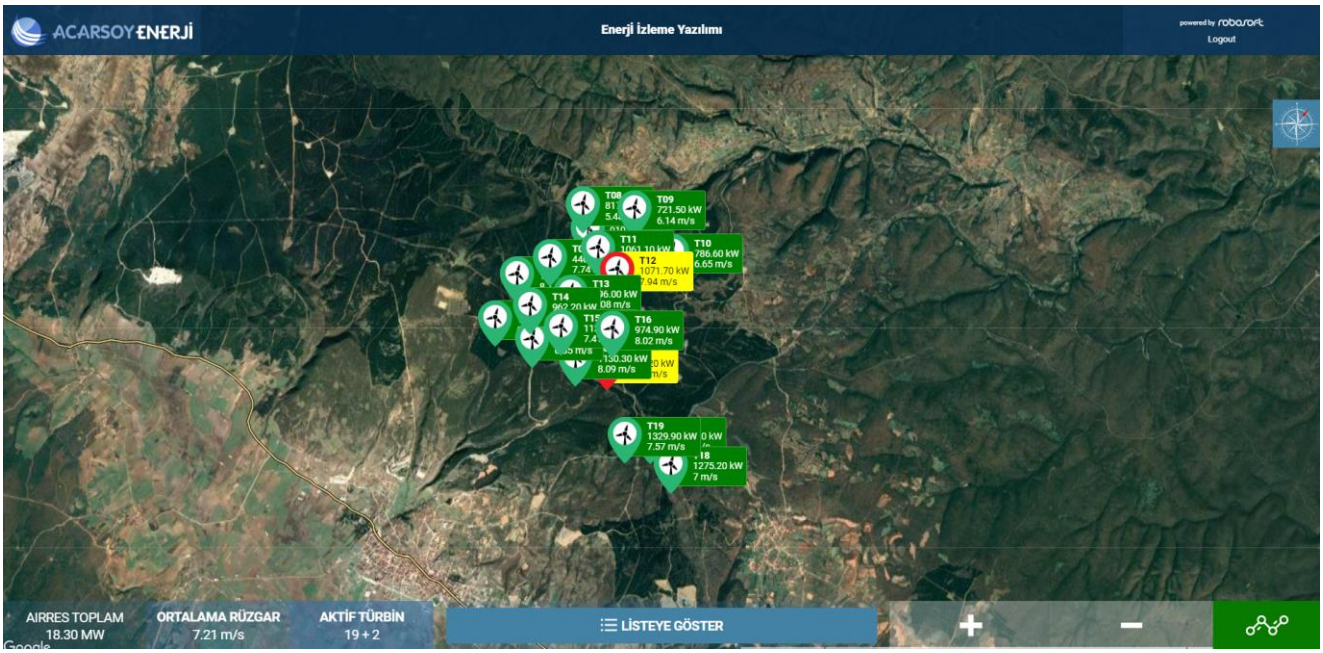

Рисунок 3.5 – Сторінка списку турбін

Якщо говорити про аналоги реалізації, то в даному випадку можна було б застосувати використання фреймворку такого як CodeIgniter або Laravel, але було прийняте рішення відмовитись від цього вибору, тому що середня навантаженість на проект досить невелика і не потребує обов’язкового використання фреймворків. Кастомна архітектрура в даному проекті написана на чистій мові PHP має в собі ті ж особливі переваги для написання невеликих проектів, що і фреймворки.

3.2 Вибір СУБД

Для статичних даних в проекті викоритовується MySQL база даних, для відображення динамічних даних у зв’язці з мовою програмуваннння JavaScript використовується MSSQL.

MySQL — це система керування реляційною базою даних SQL з відкритим вихідним кодом, яка розроблена та підтримується Oracle. MySQL спочатку був запущений у далекому 1995 році. Відтоді він пройшов через кілька змін у власності/управлінні, перш ніж опинитися в корпорації Oracle у 2010 році. Хоча Oracle зараз керує, MySQL все ще залишається програмним забезпеченням

з відкритим кодом , це означає, що ви можете вільно використовувати та змінювати його.

Рисунок 3.6 – Логотип бази даних MySQL

MySQL є важливим компонентом корпоративного стека з відкритим кодом під назвою LAMP. LAMP — це платформа веб-розробки, яка використовує Linux як операційну систему, Apache як веб-сервер, MySQL як систему керування реляційною базою даних і PHP як об'єктно-орієнтовану мову сценаріїв. Сьогодні MySQL є RDBMS, що стоїть за багатьма найкращими веб-сайтами в світі та незліченною кількістю корпоративних веб-додатків, включаючи Facebook, Twitter і YouTube. MySQL базується на моделі клієнт-сервер.

Ядром MySQL є сервер MySQL, який обробляє всі інструкції (або команди) бази даних. Сервер MySQL доступний як окрема програма для використання в мережевому середовищі клієнт-сервер і як бібліотека, яку можна вбудовувати (або пов'язувати) в окремі програми[13].

MySQL працює разом із кількома допоміжними програмами, які підтримують адміністрування баз даних MySQL. Команди надсилаються на MySQL Server через клієнт MySQL, який інстальовано на комп'ютері. MySQL спочатку був розроблений для швидкої обробки великих баз даних. Хоча MySQL зазвичай встановлюється лише на одній машині, він може надсилати базу даних у декілька

місць, оскільки користувачі мають доступ до неї через різні клієнтські інтерфейси MySQL. Ці інтерфейси надсилають оператори SQL на сервер, а потім відображають результати.

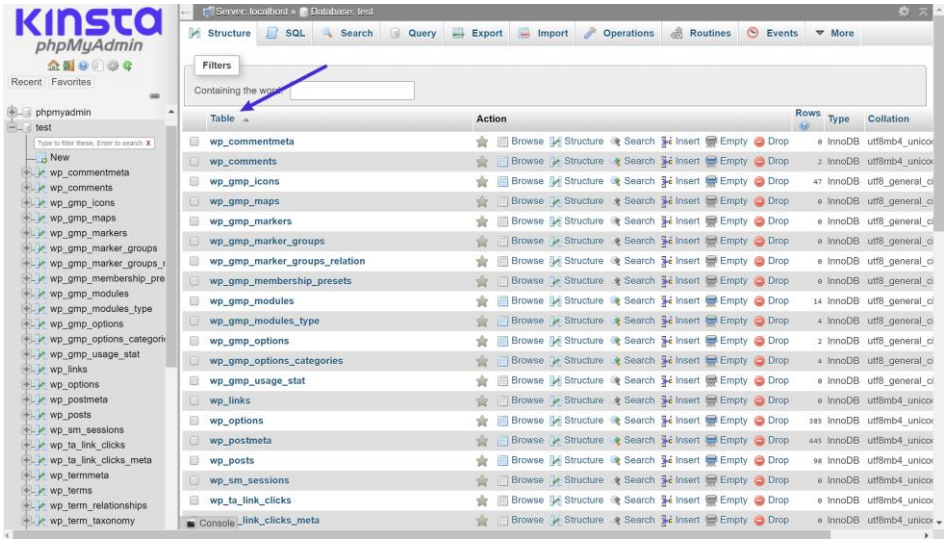

Рисунок 3.7 – Адмін-панель бази даних MySQL

MySQL дає змогу зберігати та отримувати доступ до даних у кількох механізмах зберігання, включаючи InnoDB, CSV та NDB. MySQL також здатний копіювати дані та розділяти таблиці для кращої продуктивності та довговічності. Користувачам MySQL не потрібно вивчати нові команди; вони можуть отримати доступ до своїх даних за допомогою стандартних команд SQL.

Переваги:

- забезпечує можливість запуску клієнтів і сервера на одному комп’ютері або на різних комп’ютерах через Інтернет або локальну мережу;
- має унікальну архітектуру механізму зберігання, що робить його швидшим, дешевшим і надійнішим;
- дає розробникам більшу продуктивність завдяки використанню представлень, тригерів і збережених процедур;

- простий і легкий у використанні. Ви можете створювати та взаємодіяти з MySQL, маючи лише базові знання MySQL і кілька простих операторів SQL;
- має архітектуру клієнт-сервер. Може бути будь-яка кількість клієнтів або прикладних програм, які взаємодіють із сервером бази даних (MySQL) для запиту даних, збереження змін тощо.

Недоліки:

- не дуже ефективний при обробці дуже великих баз даних.
- не має такого хорошого інструменту розробки та налагодження, ніж платні бази даних;
- Версії MySQL менше 5.0 не підтримують COMMIT, збережену процедуру та ROLE;
- схильний до пошкодження даних, оскільки він неефективний в обробці транзакцій;
- не підтримує обмеження перевірки SQL.

MSSQL Server — це система управління реляційними базами даних (RDBMS), розроблена Microsoft. Реляційна база даних заснована на архітектурі реляційної моделі. Дані організовані в таблиці (зв'язки), а таблиці пов'язані одна з одною. Кожна таблиця має рядки та стовпці (атрибути). Microsoft SQL Server є однією з основних систем керування реляційними базами даних на ринку, яка обслуговує широкий спектр програмних додатків для бізнес-аналітики та аналізу в корпоративних середовищах. Заснований на мові Transact-SQL, він містить набір стандартних мовних розширень програмування, а його додаток доступний для використання як локально, так і в хмарі.

Рисунок 3.8 – Логотип бази данних MSSQL

Microsoft SQL Server ідеально підходить для зберігання всієї потрібної інформації в реляційних базах даних, а також для безпроблемного керування такими даними завдяки своєму візуальному інтерфейсу та наявним у нього параметрам і інструментам.

Якщо у вас є список клієнтів, каталог продуктів або навіть великий вибір мультимедійного вмісту, Microsoft SQL Server допоможе керувати абсолютно всім. Це важливо для належного функціонування веб-сайту чи будь-якої програми. Його основний компонент складається з реляційного механізму, який відповідає за обробку команд, запитів, а також зберігання файлів, bb.dd., таблиць і буферів даних. Його вторинні рівні призначені для керування пам’яттю, програмування та адміністрування взаємодії запитів і відповідей із серверами, на яких розміщено бази даних[13].

Переваги:

- Підвищений рівень безпеки даних
- Простота налаштування
- Оптимізоване зберігання даних
- Підтримка відновлення даних

Недоліки:

- Вартість
- Обмежена сумісність

- Обмеження апаратного забезпечення

3.3 Опис коду

Основний функціонал проекту та сама веб-сторінка знаходиться на серверній частині, тому більшість опису коду буде стосуватись саме цієї частини. На рисунку 3.9 зображена організація файлів проекту, що буде розібрана в цьому пункті.

Код програми буде наведений в додатку даної магістерської роботи. Для кращого розуміння проекту буде розібрана послідовна робота кожної з частин проекту. Для початку обираємо функціональність головної частини проекту. В ньому буде описане як відбувається відкриття головної сторінки веб-сайту та детальний опис всіх його частин. Головна сторінка відкривається відразу після авторизації. За її функціональність відповідає функція _main, що грає роль контроллеру в проекті.

```
if (!$output->user['user_id'])
    HTTP::redirect(TAK_URL . '/login');
```

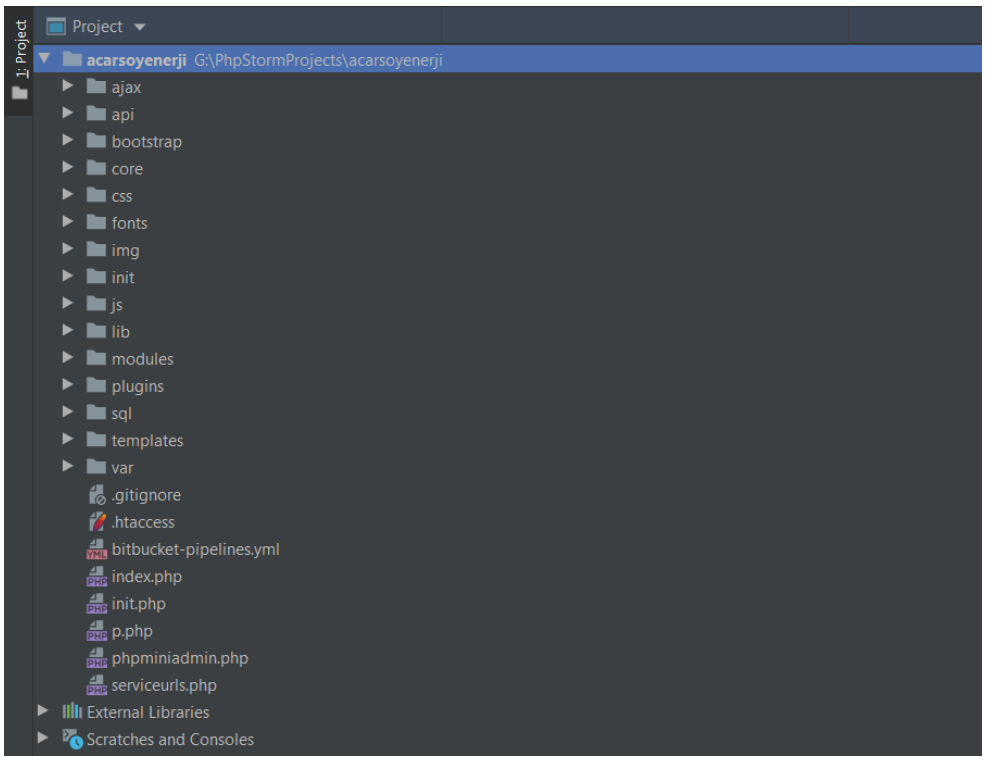

Рисунок 3.9 – Організація файлів серверної частини

Даний блок коду перевіряє наявність параметру user_id об’єкту user. Якщо цей параметр відсутній, відбувається перенесення користувача на сторінку авторизації.

```
AV_USEFULL::UpdateSession();
$request = GPCF::get_instance();
$db = DB::get_instance();
```

Тут відбувається оновлення сесії, та оголошення об’єктів запиту та бази даних. Саме ці об'єкти будуть надалі використовуватись для доступу до запитів або баз даних.

```
require_once 'ajax/map-data.php';
$output->facilities = GetPlants();
```

Далі виконуємо імпортування всіх функцій з шляху ajax/map-data.php і після цього виконуємо імпортовану функцію GetPlants() та збереження його результату в об’єкт facilities. Далі поговоримо детальніше про вміст функції GetPlants().

```
$db = DB::get_instance();
$now = time();
$cache = $db->get_row( "SELECT * FROM `cache` WHERE turbine_id=-1" );
if( ($now-$cache['cached']) < 60 && $getplanttype=='ALL' ){
  return $cache['response'];
}
```

В цьому блоці коду відбувається збір даних кешу і перевірка на те, чи не пройшов час зберігання коду. Якщо кількість зберігання кешу не перевищило 60 хвилин, тоді повертається закешовані дані. Це було зроблено для економії часу та пам’яті на завантаження даних турбін.

```
$res = array();
$fieldsMap = array();
$mssql = MSSQLDB::get_instance();
if( ! $mssql )
    die( '~error~' );
$rated_power = array();
```

Тут відбувається ініціалізація параметрів початкових параметрів та об’єкта бази даних MSSQL. В подальшому ці параметри будуть використовуватись для збереження відповіді, зберігання полів певного бренду та розрахунку ефективності турбін.

```
$data = $db->get_all( "SELECT * FROM brands");
for( $i=0; $i<count($data); $i++)
{
   $rated_power[ $data[$i]['id'] ] = $data[$i]['rated_power'];
}
```

За рахунок все ініціалізованих параметрів тут відбувається калькуляція суми ефективності виробництва електроенергії.

```
$plants = $db->get_all( "SELECT * FROM plants" );
for( $i=0; $i<count($plants); $i++ )
{
   $turbines = $db->get_all( "SELECT * FROM turbines WHERE
plant_id={$plants[$i]['id']}" );


$plants[$i]['mapname'] ? $plants[$i]['mapname'] :
$plants[$i]['name'],
   'type' => strtolower($plants[$i]['type']),
   'lat' => $plants$plant = array(
   'id' => strtolower($plants[$i]['name']),
   'name' => [$i]['latitude'],
   'lng' => $plants[$i]['longitude'],
   'center' => array('lat' => $plants[$i]['avg_latitude'], 'lng' =>
$plants[$i]['avg_longitude'])
);
```

В даному блоці коду відбувається збірка всіх плантацій, і за цими плантаціями збірка всіх турбін. В останній частині коду відбувається форматування масиву плантацій для більш зручної роботи з ним в подальшому. В наступних блоках коду відбувається зчитування полів певного бренду для подальної роботи з динамічними даними, а після цього вже безпосередньо робота з даними. Варто зауважити, специфіка збору назв полів та відповідно збір за ними даних відбувається залежно від певного типу турбін.

```
$resTxt = json_encode($res);
   if( $getplanttype=='ALL' )
   {
      $sql = "REPLACE INTO cache (`turbine_id`,`cached`,`response`)
VALUES (-1,%l,'%s')";
      $db->query( $sql, array($now, $resTxt) );
   }
   return $resTxt;
```

Під кінець файлу відбувається збереження кешу, та форматування відповіді функції у потрібний формат для подальшої роботи з ним в JavaScript коді. Надалі повернемось до нашої головної функції _main в якій міститься основний функціонал відкриття головної сторінки проекту.

```
$output->initialZoom = 6;
$output->defaultLat = 0;
$output->defaultLng = 0;
$output->centerLat = 0;
$output->centerLng = 0;
$arr = json_decode($output->facilities, true);
if (count($arr)) {
    for ($i = 0; $i < count($arr); $i++) {
        $output->defaultLat += $arr[$i]['lat'];
        $output->defaultLng += $arr[$i]['lng'];
    }
    $output->defaultLat /= count($arr);
    $output->defaultLng /= count($arr);
}

$plant_id = $request->get('plant_id');
if( $plant_id ) {
    $plant = $db->get_row("SELECT * FROM plants WHERE `id`=%l",
array($plant_id));
    if( $plant ) {
        $output->centerLat = $plant['avg_latitude'];
        $output->centerLng = $plant['avg_longitude'];
        $output->initialZoom = 13;
        $output->plant_id = strtolower($plant['name']);
    }
}
```

Тут відбувається налаштування початкових параметрів розташування карти по дефолту та зуму. Саме вони надалі, в JavaScript коді будуть використовуватись для налаштування карти.

3.4 Результати тестування сервісу

Перевірка працездатності проекту відбувається у декілька етапів. Послідовність тестування буде аналогічна до того, як відбувається взаємодія користувача с сервісом. З самого початку користувач стикається з функціоналом авторизації. Для авторизації користувачу необхідно ввести логін та пароль, що зображено на рисунку 3.10.

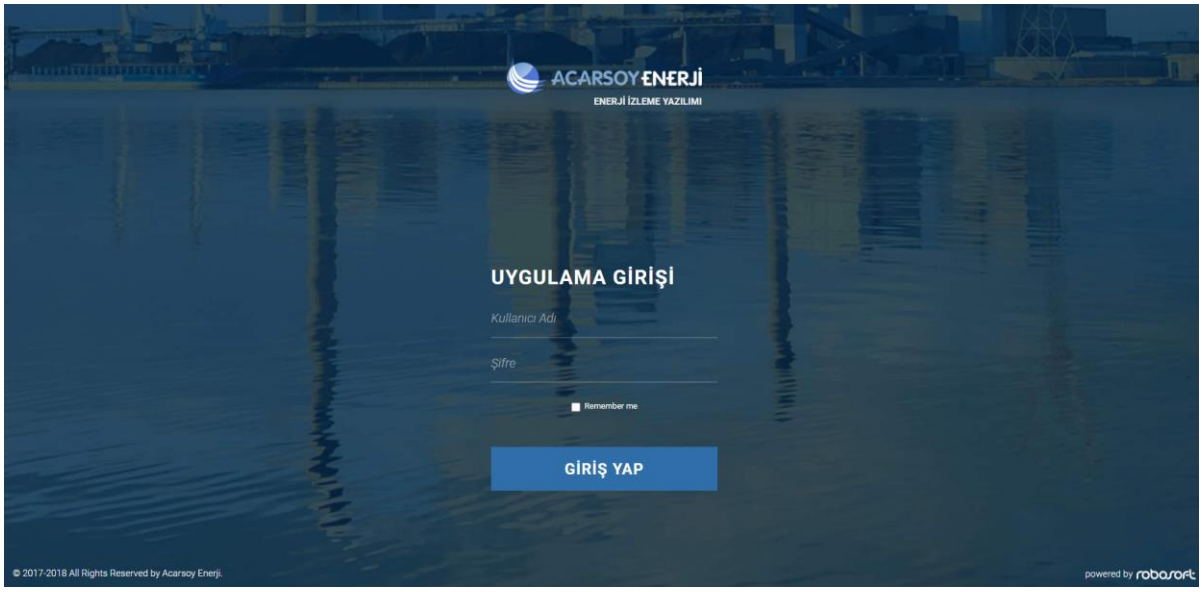

Рисунок 3.10 – Сторінка авторизації користувача

Якщо користувач вводить невірний логін і пароль, користувачу у верхньому правому куті відображується повідомлення Incorrect email or password. Якщо користувач вводить вірний логін і пароль, користувач потрапляє на головну сторінку сервісу.

Рисунок 3.11 – Сторінка авторизації користувача з повідомленням про неуспішну авторизацію

Так само, як і для звичайного користувача існує авторизація для користувачів з правами admin. Логіка роботи даної сторінки є ідентичною до панелі авторизації для звичайних користувачів.

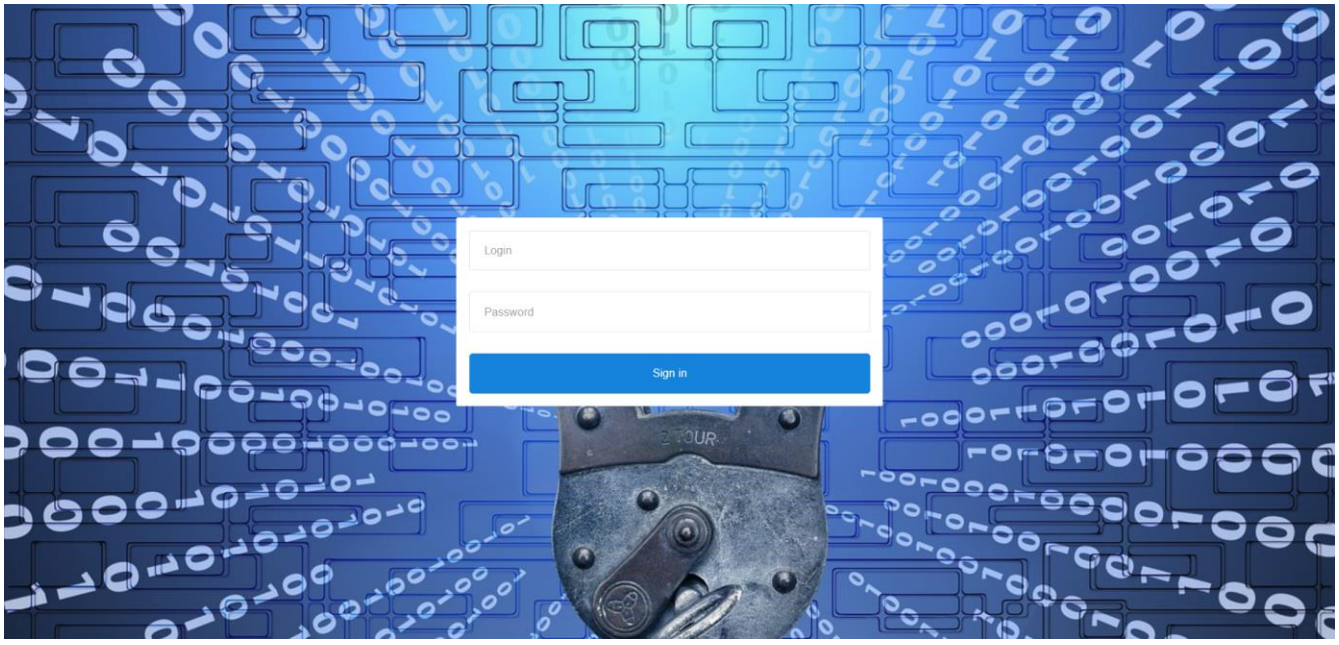

Рисунок 3.12 – Сторінка авторизації користувача з правами admin

Для авторизації користувач повинен ввести логін і пароль. Якщо він введений вірно, користувач потрапляє на головну сторінку admin панелі. Якщо логін і пароль не вірний, користувач бачить повідомлення Error! Incorrect login or password.

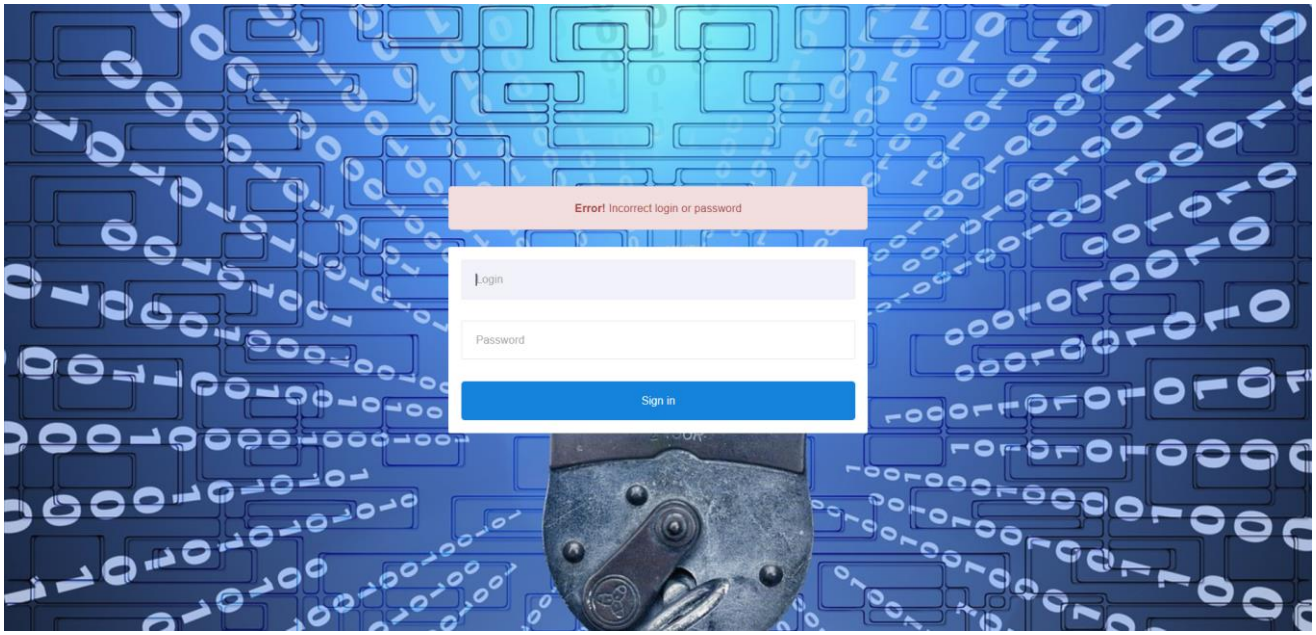

Рисунок 3.13 – Сторінка авторизації користувача з правами admin з повідомленням про неуспішну авторизацію

При нормальній роботі проекту на головній сторінці повинні відображуватись плантації, які прописані в базі даних. Для тестування головної сторінки скористаємось адмін-панеллю проекту, що використовується для збереження даних. Плантації заповнюємо на сторінці додавання плантацій, що зображена на рисунку 3.14.

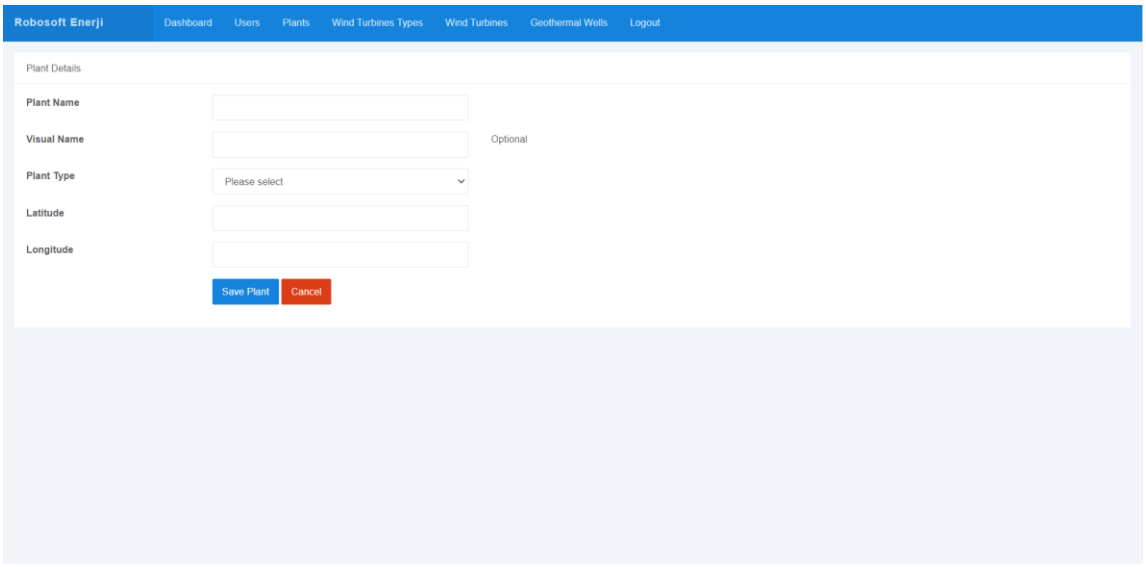

Рисунок 3.14 – Сторінка додавання плантації адмін-панелі

Після заповнення даних, можемо бачити на головній сторінці плантації, що ми створили на рисунку 3.15.

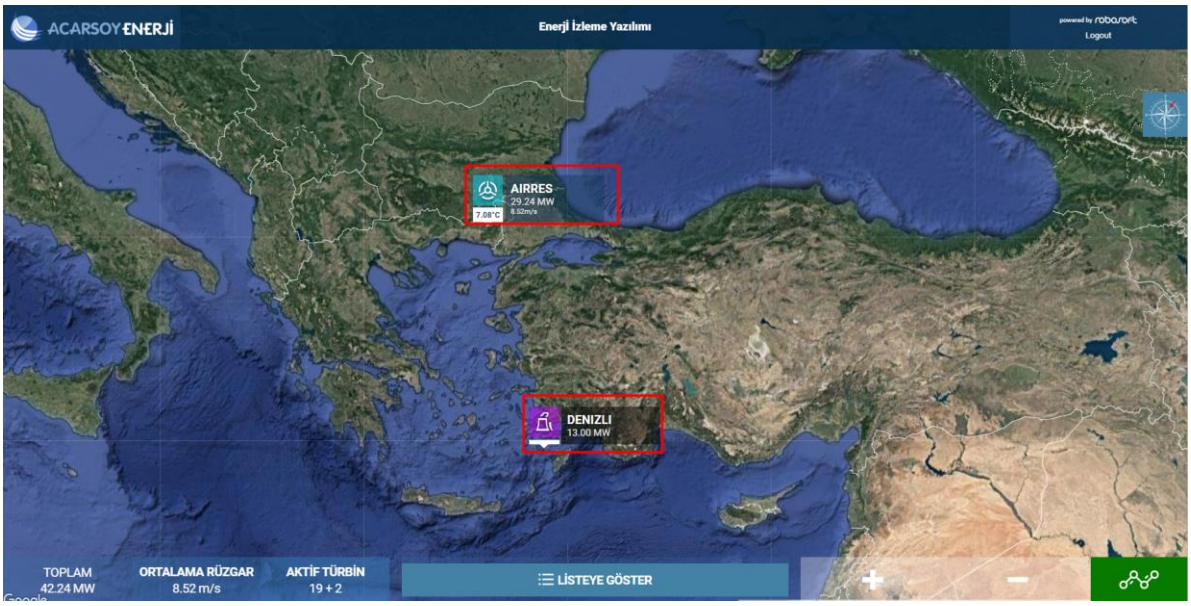

Рисунок 3.15 – Головна сторінка сервісу

Після заповнення даних плантацій треба заповнити бренд плантації, що буде в собі зберігати колонки даних турбін, які використовуються для збору даних в динамічній базі даних SQL Server.

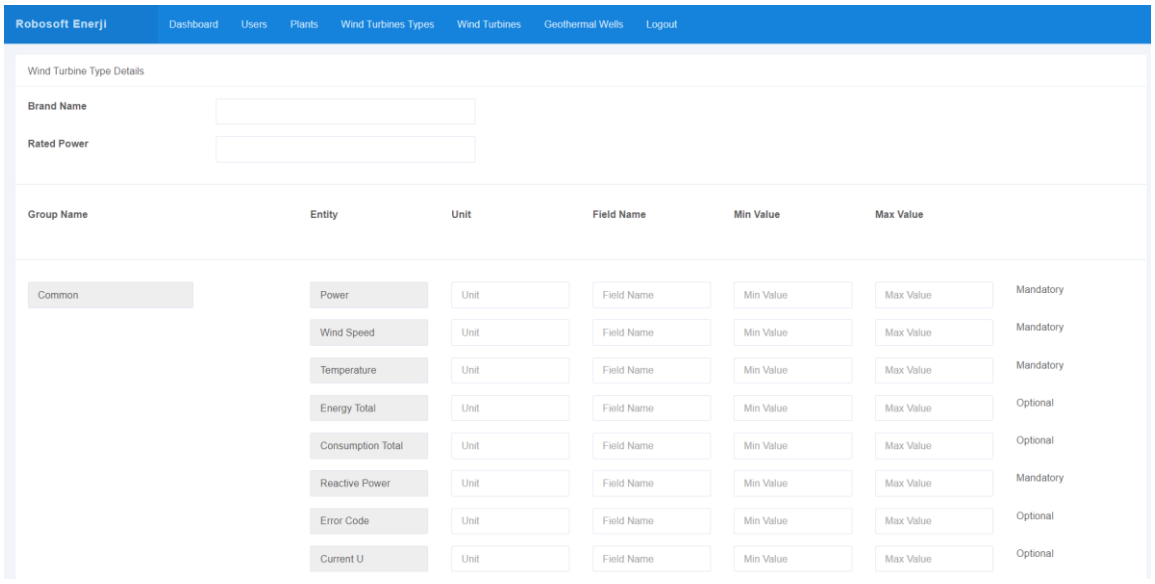

Рисунок 3.16 – Сторінка додавання бренду плантації

Після додавання бренду плантації можемо зайнятись безпосередньо заповненням даних турбін плантації, використовуючи дані відповідної плантації та бренду, що відноситься до турбіни.

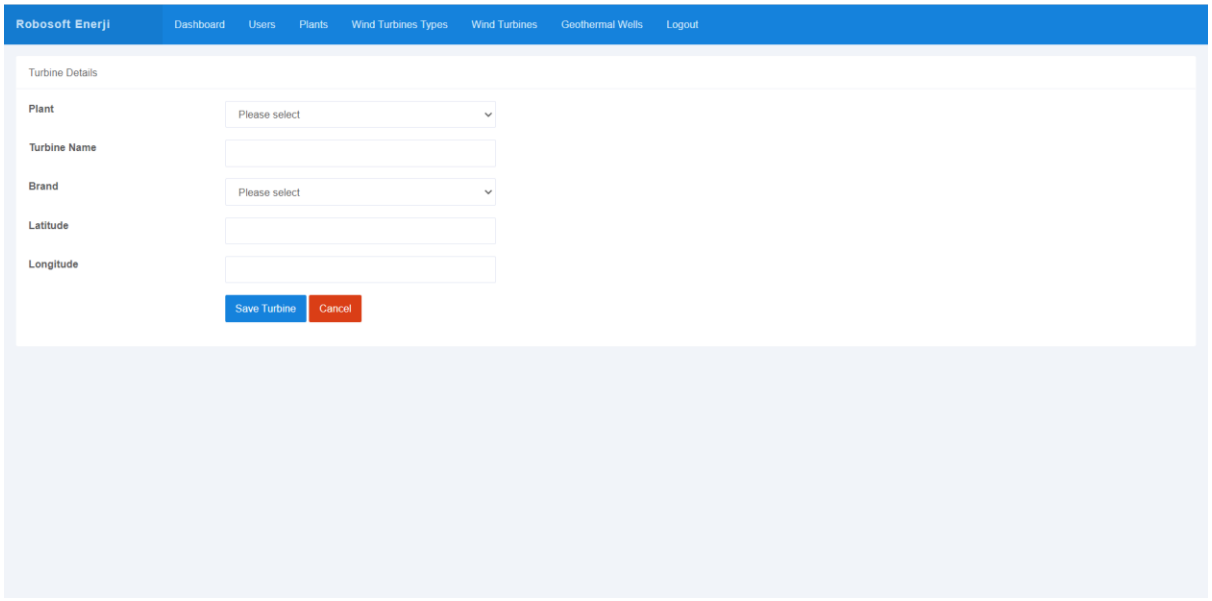

Рисунок 3.17 – Сторінка додавання турбіни плантації

Після заповнення даних в адмін-панелі та відповідному заповненню даних в базі даних SQL Server, можемо бачити список турбін плантації на рисунку 3.18.

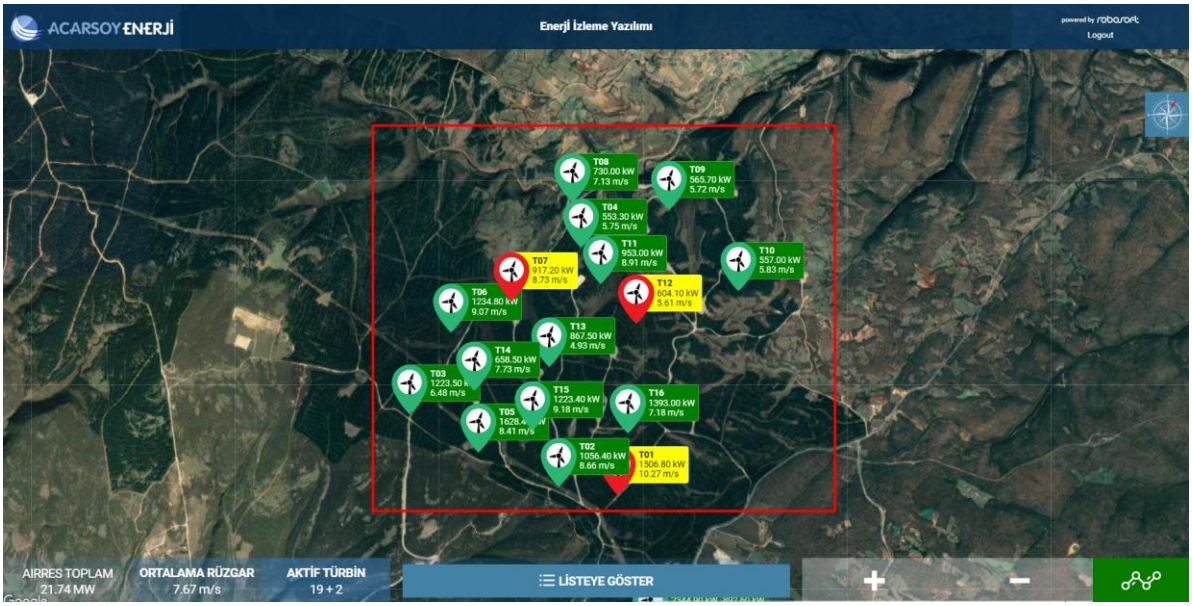

Рисунок 3.18 – Сторінка списку турбін плантації

Натиснувши на будь-яку із турбін можемо отримати список детальних даних зображених на рисунку 3.19.

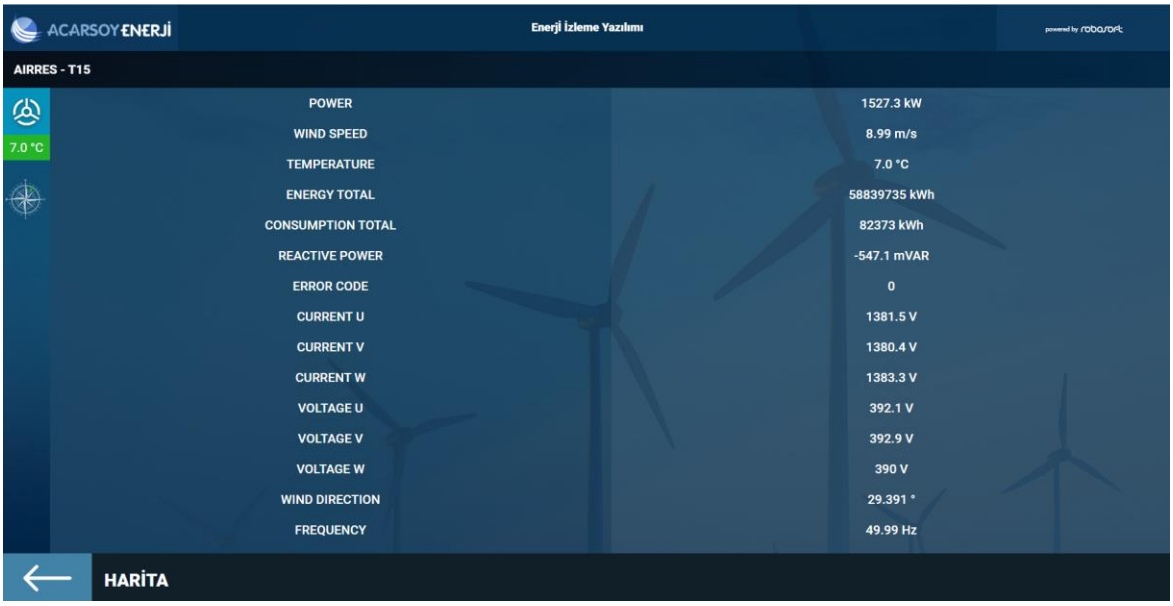

Рисунок 3.19 – Сторінка деталей турбіни

ВИСНОВКИ

Під час виконання кваліфікаціної роботи магістра було розроблено інформаційно-аналітичну технологію моніторингу виробництва електроенергії, що дає можливість цілодобово наживо спостерігати за усіма найважливішими параметрами електростанцій, включаючи кількість виробленої електроенергії, так і параметри, які відповідають за стан кожної турбіни. При цьому виконані такі основні завдання:

1. Виконано інформаційно-аналітичний огляд сучасних сервісів моніторингу статистики електростанцій.

2. Розроблено інформаційну модель інформаційно-аналітичної технології моніторингу виробництва електроенергії

3. Розроблено і нормалізовано до третьої нормальної форми структуру бази даних з 12 таблиць

4. Засобами програмної реалізації було обрано мови програмування PHP та JavaScript. При реалізації баз даних в ролі СУБД були обрані MySQL та SQL Server

5. З використанням розробленої інформаційно-аналітичнлї технології моніторингу виробництва електроенергії спроєктовано і програмно реалізовано веб-сервіс моніторингу статистики електростанцій.

6. Перевірено працездатність розробленого веб-сервісу.

СПИСОК ЛІТЕРАТУРИ

1. David Carr, Markus Gray. Beginning PHP: Master the latest features of PHP 7 and fully embrace modern PHP development. – Birmingham: Packt Publishing, 2018. - 214 с.

2. Steve Prettyman. Learn PHP 8: Using MySQL, JavaScript, CSS3, and HTML5 – New York City: Apress, 2020. - 452 с.

3. Kevin Tatroe. Programming PHP: Creating Dynamic Web Pages. – California: O`relly Media, 2020. - 544 с.

4. Rob Foster. CodeIgniter Web Application Blueprints. – Birmingham: Packt Publishing, 2015. - 330 с.

5. Luke Welling, Laura Thomson. PHP and MySQL Web Development (Developer`s Library) 5th Edition. – Boston: Addison-Wesley Professional, 2016. – 688 с.

6. Adam Omelak. Laminas: MVC Framework for PHP. – Birmingham: Packt Publishing, 2020. – 466 с.

7. Daniel Nichter. Efficient MySQL Performance: Best Practices and Techniques 1st Edition. - California: O'Reilly Media, 2022. - 335 с.

8. Dmitri Korotkevitch. SQL Server Advanced Troubleshooting and Performance Tuning. 1st Ed. - California: O'Reilly Media, 2022. - 500 с.

9. David Flanagan. JavaScript: The Definitive Guide: Activate Your Web Pages. - California: O`relly Media, 2015. - 1093 с.

10. Anthony Molinaro, Robert de Graaf. SQL Cookbook. 2nd Ed. - California: O`relly Media, 2020. – 500 с.

11. Marijn Haverbeke. Eloquent JavaScript, 3rd Edition: A Modern Introduction to Programming 3rd Edition. – San Francisco: No Starch Press, 2018. – 472 с.

12. Robin Nixon. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. - California: O`relly Media, 2014. – 812 с.

13. Itzik Ben-Gan. T-SQL Fundamentals (Developer Reference) 3rd Edition. – Redmond: Microsoft Press, 2016. – 464 с.

14. Josh Lockhart. Modern PHP: New Features and Good Practices 1st Edition. - California: O`relly Media, 2015. – 270 с.

15. Larry Ullman. PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide. – California: Peachpit Press, 2014. – 696 с.

ДОДАТОК

Код файлу index.php:

```
<?php

class index extends Module {

    public function init() {
        $this->name = 'index';
        $this->actions = array(
            'main' => array('main'),
            'login' => array('login'),
            'logout' => array('logout'),
            'turbines' => array('turbines'),
            'turbine' => array('turbine'),
            'trendler' => array('trendler'),
        );

        $this->init_tables();
    }

    public function validate(GPCF $request, Output $input) {
        $this->validated = true;
        $this->set_default_action('main');
    }

    /*
      ----
     */

    protected function _main($input, $output) {
        if (!$output->user['user_id'])
            HTTP::redirect(TAK_URL . '/login');

        AV_USEFULL::UpdateSession();

        $request = GPCF::get_instance();
        $db = DB::get_instance();

        require_once 'ajax/map-data.php';
        $output->facilities = GetPlants();

        $output->initialZoom = 6;
        $output->defaultLat = 0;
        $output->defaultLng = 0;
        $output->centerLat = 0;
```

```
        $output->centerLng = 0;

        $arr = json_decode($output->facilities, true);
        if (count($arr)) {
            for ($i = 0; $i < count($arr); $i++) {
                $output->defaultLat += $arr[$i]['lat'];
                $output->defaultLng += $arr[$i]['lng'];
            }
            $output->defaultLat /= count($arr);
            $output->defaultLng /= count($arr);
        }

        $plant_id = $request->get('plant_id');
        if( $plant_id ) {
            $plant = $db->get_row("SELECT * FROM plants WHERE
`id`=%l", array($plant_id));
            if( $plant ) {
                $output->centerLat = $plant['avg_latitude'];
                $output->centerLng = $plant['avg_longitude'];
                $output->initialZoom = 13;
                $output->plant_id = strtolower($plant['name']);
            }
        }

        $output->useCompassCSS = true;
        $output->title = 'Main screen';
        $output->template = 'main.tpl';
    }

    /*
      ----
     */

    function _login($input, $output) {
        if ($output->user['user_id'])
            HTTP::redirect(TAK_URL);

        $request = GPCF::get_instance();
        $db = DB::get_instance();

        $op = $request->get('op');
        $email = $request->get('email');
        $password = $request->get('password');
        $remember = ($request->get('remember') == 'on');

        if ($op == 'login' && $email && $password) {
```

```
            $user = $db->get_row("SELECT * FROM `users` WHERE
email='%s' AND password='%s'", array($email, $password));
            if ($user['id']) {
                $user['user_id'] = $user['id'];
                AV_USEFULL::CreateSessionVars($user);

                if ($remember) {
                    $tm = time() + 60 * 60 * 24 * 30;
                    setcookie('usreml', md5($email), $tm);
                    setcookie('usrpwd', md5($password), $tm);
                }

                HTTP::redirect(TAK_URL);
            } else
                TAK::message('Incorrect email or password');
        }

        $output->loginbackground = true;
        $output->title = 'Login';
        $output->template = 'login.tpl';
    }

    /*
      ----
     */

    function _logout($input, $output) {
        if ($output->user['user_id'])
            AV_USEFULL::ClearSessionCookies();

        HTTP::redirect(TAK_URL . '/login');
    }

    /*
      ----
     */

    function _turbines($input, $output) {
        if (!$output->user['user_id'])
            HTTP::redirect(TAK_URL);

        AV_USEFULL::UpdateSession();

        $request = GPCF::get_instance();
        $db = DB::get_instance();
```

```
        $id = $request->get('id');

        $output->plant = $db->get_row("SELECT * FROM plants WHERE
`id`=%l", array($id));
        if (!$output->plant['id']) {
            TAK::message("Unknown plant: $id");
            HTTP::redirect(TAK_URL);
        }

        $output->turbinesbackground = true;
        $output->title = $output->plant['name'];
        $output->template = 'turbines.tpl';
    }

    /*
      ----
     */

    function _turbine($input, $output) {
        if (!$output->user['user_id'])
            HTTP::redirect(TAK_URL);

        AV_USEFULL::UpdateSession();

        $request = GPCF::get_instance();
        $db = DB::get_instance();

        $plantname = $request->get('id');
        $turbinename = $request->get('id2');

        $output->plant = $db->get_row("SELECT * FROM plants WHERE
`name`='%s'", array($plantname));
        if (!$output->plant['id']) {
            TAK::message('Unknown plant ' . $plantname);
            HTTP::redirect(TAK_URL);
        }

        $output->turbine = $db->get_row("SELECT * FROM turbines WHERE
`name`='%s' AND plant_id=%l", array($turbinename, $output-
>plant['id']));
        if (!$output->turbine['id']) {
            TAK::message("Unknown turbine $turbine_id at plant
$plantname");
            HTTP::redirect(TAK_URL);
        }
```

```
        $output->turbinesbackground = true;
        $output->useCompassCSS = true;
        $output->title = $output->plant['name'];
        $output->template = 'turbine.tpl';
    }

    /*
      ----
     */

    function _trendler($input, $output) {
        if (!$output->user['user_id'])
            HTTP::redirect(TAK_URL);

        AV_USEFULL::UpdateSession();

        $request = GPCF::get_instance();
        $db = DB::get_instance();

        $id = $request->get('id');

        $output->plant = $db->get_row("SELECT * FROM plants WHERE
`id`=%l", array($id));
        if (!$output->plant['id']) {
            TAK::message("Unknown plant: $id");
            HTTP::redirect(TAK_URL);
        }

        $output->titleToday = date('d.m.Y');
        $output->titleWeek = date('d.m.Y', strtotime('last monday'))
. ' - ' . date('d.m.Y');
        $output->titleMonth = '1.' . date('m') . '.' . date('Y') . '
- ' . date('d.m.Y');
        $output->titleYear = '1.01.' . date('Y') . ' - ' .
date('d.m.Y');

        $output->title = $output->plant['name'];
        $output->template = 'trendler.tpl';
    }

}
```

Код файлу map-data.php:

```
<?php

if( !defined('TAK_DIR') )
```

```
   require_once '../init.php';

function GetPlants($getplanttype='ALL')
{
   $db = DB::get_instance();
   $now = time();
   $cache = $db->get_row( "SELECT * FROM `cache` WHERE turbine_id=-1"
);
   if( ($now-$cache['cached']) < 60 && $getplanttype=='ALL' )
   {
      return $cache['response'];
   }

   $res = array();
   $fieldsMap = array();

   $mssql = MSSQLDB::get_instance();
   if( ! $mssql )
      die( '~error~' );

   $rated_power = array();

   $data = $db->get_all( "SELECT * FROM brands");
   for( $i=0; $i<count($data); $i++)
   {
      $rated_power[ $data[$i]['id'] ] = $data[$i]['rated_power'];
   }

   $plants = $db->get_all( "SELECT * FROM plants" );
   for( $i=0; $i<count($plants); $i++ )
   {
      $turbines = $db->get_all( "SELECT * FROM turbines WHERE
plant_id={$plants[$i]['id']}" );
      switch( $plants[$i]['type'] )
      {
         case 'WIND':
         case 'GEO':
            if( count($turbines) == 0 )
                  continue;
      }

      $plant = array(
         'id' => strtolower($plants[$i]['name']),
         'name' => $plants[$i]['mapname'] ? $plants[$i]['mapname'] :
$plants[$i]['name'],
         'type' => strtolower($plants[$i]['type']),
```

```
         'lat' => $plants[$i]['latitude'],
         'lng' => $plants[$i]['longitude'],
         'center' => array('lat' => $plants[$i]['avg_latitude'],
'lng' => $plants[$i]['avg_longitude'])
      );
      if( $plants[$i]['type'] == 'WIND' && ($getplanttype=='ALL' ||
$getplanttype=='WIND') )
      {
         $sql = '';
         $arrParams = array();
         $arrCoord = array();
         $plant_rated_power = 0;

         for( $j=0; $j<count($turbines); $j++ )
         {
            $plant_rated_power += $rated_power[
$turbines[$j]['brand_id'] ];
            if( ! isset($fieldsMap[$turbines[$j]['brand_id']]) )
            {
               $group_id = $db->get_one( "SELECT id FROM
brand_field_groups WHERE brand_id={$turbines[$j]['brand_id']} ORDER
BY id LIMIT 1" );
               $fields = $db->get_all( "SELECT `name`,`field` FROM
brand_fields WHERE group_id=$group_id" );
               $arr = array();
               for( $k=0; $k<count($fields); $k++ )
               {
                  $arr[$fields[$k]['name']] = $fields[$k]['field'];
               }
               $fieldsMap[$turbines[$j]['brand_id']]  = $arr;
            }
            if( $sql )
               $sql .= ' UNION ALL ';
            $sql .= "SELECT '{$turbines[$j]['name']}' AS [Name],
[RealDate]"
               . ",{$fieldsMap[$turbines[$j]['brand_id']]['Power']}
AS Power"
               . ",{$fieldsMap[$turbines[$j]['brand_id']]['Wind
Speed']} AS WindSpeed"
               .
",{$fieldsMap[$turbines[$j]['brand_id']]['Temperature']} AS
Temperature"
               . ",{$fieldsMap[$turbines[$j]['brand_id']]['Reactive
Power']} AS ReactivePower"
            ;
```

```
            if( $fieldsMap[$turbines[$j]['brand_id']]['Energy
Total'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Energy Total']}";
            else
               $sql .= ",0";
            $sql .= "  AS EnergyTotal";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Consumption
Total'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Consumption Total']}";
            else
               $sql .= ",0";
            $sql .= "  AS ConsumptionTotal";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Error Code'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Error Code']}";
            else
               $sql .= ",-1";
            $sql .= "  AS dbError";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Current U'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Current U']}";
            else
               $sql .= ",0";
            $sql .= "  AS CurrentU";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Current V'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Current V']}";
            else
               $sql .= ",0";
            $sql .= "  AS CurrentV";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Current W'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Current W']}";
            else
               $sql .= ",0";
            $sql .= "  AS CurrentW";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Voltage U'])
               $sql .=
```

```
",{$fieldsMap[$turbines[$j]['brand_id']]['Voltage U']}";
            else
               $sql .= ",0";
            $sql .= "  AS VoltageU";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Voltage V'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Voltage V']}";
            else
               $sql .= ",0";
            $sql .= "  AS VoltageV";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Voltage W'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Voltage W']}";
            else
               $sql .= ",0";
            $sql .= "  AS VoltageW";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Wind
Direction'])
               $sql .= ",{$fieldsMap[$turbines[$j]['brand_id']]['Wind
Direction']}";
            else
               $sql .= ",0";
            $sql .= "  AS WindDirection";

            if( $fieldsMap[$turbines[$j]['brand_id']]['Frequency'])
               $sql .=
",{$fieldsMap[$turbines[$j]['brand_id']]['Frequency']}";
            else
               $sql .= ",0";
            $sql .= "  AS Frequency";

            $sql .= " FROM (SELECT TOP 1 * FROM %s ORDER BY ID DESC)
t{$turbines[$j]['name']}";
            $arrParams[] = $plants[$i]['name'] . '_' .
substr($plants[$i]['type'], 0, 1) . $turbines[$j]['name'];

            $arrCoord[$turbines[$j]['name']] = array('lat' =>
$turbines[$j]['latitude'], 'lng' => $turbines[$j]['longitude']);
         }

         $wind = 0;
            $windDirection = 0;
         $temperature = 0;
```

```
        $cntNoError = 0;

        $data = $mssql->get_all( $sql, $arrParams );
        for( $j=0; $j<count($data); $j++ )
        {
           $error = $data[$j]['dbError'];

              $plant['power'] += $data[$j]['Power'] / 1000;
              $plant['powerR'] += $data[$j]['ReactivePower'] / 1000;

              $wind += $data[$j]['WindSpeed'];
                   $windDirection += $data[$j]['WindDirection'];
              $temperature += $data[$j]['Temperature'];
           if( $error == 0 ) // 30/08/18
           {                 // 30/08/18
              $cntNoError ++;
           }

           $plant['buildings'][] = array(
              "a" => $data[$j]['Name'],
              "b" => 'wind',
              "c" => $arrCoord[$data[$j]['Name']]['lat'],
              "d" => $arrCoord[$data[$j]['Name']]['lng'],
              "e" => sprintf('%.2f', $data[$j]['Power']),
              "f" => sprintf('%.2f', $data[$j]['ReactivePower']),
              "g" => $data[$j]['WindSpeed'],
              "h" => sprintf('%.2f', $data[$j]['WindDirection']),
              "i" => $data[$j]['Temperature'],
              "j" => sprintf('%.2f', $data[$j]['CurrentU']),
              "k" => sprintf('%.2f', $data[$j]['CurrentV']),
              "l" => sprintf('%.2f', $data[$j]['CurrentW']),
              "m" => sprintf('%.2f', $data[$j]['VoltageU']),
              "n" => sprintf('%.2f', $data[$j]['VoltageV']),
              "o" => sprintf('%.2f', $data[$j]['VoltageW']),
              "u" => sprintf('%.2f', $data[$j]['Frequency']),
              "p" => intval($error),
              "plant" => strtolower($plants[$i]['name'])
           );
        }

        $plant['power'] = sprintf('%.2f', $plant['power']);
        $plant['powerR'] = sprintf('%.2f', $plant['powerR']);
        //$plant['wind'] = ($cntNoError > 0) ? sprintf('%.2f',
$wind/$cntNoError) : 0;
        $plant['wind'] = (count($turbines) > 0) ? sprintf('%.2f',
$wind/count($turbines)) : 0;
```

```
            //$plant['windDirection'] = ($cntNoError > 0) ?
sprintf('%.2f', $windDirection/$cntNoError) : 0;
         $plant['windDirection'] = (count($turbines) > 0) ?
sprintf('%.2f', $windDirection/count($turbines)) : 0;
         //$plant['temperature'] = ($cntNoError > 0) ?
sprintf('%.2f', $temperature/$cntNoError) : 0;
         $plant['temperature'] = (count($turbines) > 0) ?
sprintf('%.2f', $temperature/count($turbines)) : 0;
         $plant['efficiency'] = intval( 100 * (1000 *
$plant['power']) / $plant_rated_power );
         $plant['version'] = '2018-09-13-01';
      }

      elseif( $plants[$i]['type'] == 'GEO' && ($getplanttype=='ALL'
|| $getplanttype=='GEO') )
      {
         $plant_rated_power = 0;

         $plant['power'] = 0;
         $plant['powerR'] = 0;

         $sql = "SELECT TOP 1 * FROM {$plants[$i]['name']}_GEO ORDER
BY ID DESC";
         $data = $mssql->get_row( $sql );
         $dbTime = strtotime( $data['RealDate'] );
         $bError = (($now - $dbTime) > 2*60*60);
         {
            $plant['power'] = sprintf('%.2f', $data['Unit1Energy'] +
$data['Unit2Energy'] );
                $plant['YVoltage'] = sprintf('%.2f',
$data['YVoltage']);
                $plant['OVoltage'] = sprintf('%.2f',
$data['OVoltage']);
                $plant['GenVoltage'] = sprintf('%.2f',
$data['GenVoltage']);
                $plant['PowerFactor'] = sprintf('%.2f',
$data['PowerFactor']);
         }

         for( $j=0; $j<count($turbines); $j++ )
         {
            $plant['buildings'][] = array(
               "a" => $turbines[$j]['name'],
               "b" => 'geo',
               "c" => $turbines[$j]['latitude'],
               "d" => $turbines[$j]['longitude'],
```

```
                "e" => sprintf('%.2f',
$data["{$turbines[$j]['name']}Energy"]),
                "f" => sprintf('%.2f',
$data["{$turbines[$j]['name']}ReactiveEnergy"]),
                "p" => 0,
                "plant" => strtolower($plants[$i]['name'])
            );
            }

         $plant['efficiency'] = $plant_rated_power ? intval( 100 *
$plant['power'] / $plant_rated_power ) : 0;
      }

      $res[] = $plant;
   }

   $resTxt = json_encode($res);

   if( $getplanttype=='ALL' )
   {
      $sql = "REPLACE INTO cache (`turbine_id`,`cached`,`response`)
VALUES (-1,%l,'%s')";
      $db->query( $sql, array($now, $resTxt) );
   }
   return $resTxt;
}

switch( $_REQUEST['op'] )
{
   case 'getplants':
      header('Content-Type: application/json;charset=UTF-8');
      $session = new Session();
      if( ! Session::get('user_id' ) )
         die( json_encode(array('result'=>-1,'message'=>'You are not
logged in')) );

      AV_USEFULL::UpdateSession();
      die( GetPlants() );
}
```

Код файлу map.js:

```
var map = null;
var turbines = null;
var currentWellId = -1;
var totalallPower = 0;
var defaultZoom = 6;
```

```
function InitMap(google) {
    if ($('#map').length) {
        var zoom = initialZoom;
        if (!zoom)
            zoom = defaultZoom;
        var lat = initLat;
        if (!lat)
            lat = defaultLat;
        var lng = initLng;
        if (!lng)
            lng = defaultLng;
        map = new google.maps.Map(document.getElementById('map'), {
            center: {lat: lat, lng: lng},
            zoom: zoom,
            minZoom: 5,
            maxZoom: 17,
            disableDefaultUI: true,
            mapTypeId: google.maps.MapTypeId.HYBRID,
            styles: [
                {"elementType": "labels", "stylers": [{"visibility":
"off"}]},
                {"featureType": "administrative.land_parcel",
"stylers": [{"visibility": "off"}]},
                {"featureType": "administrative.neighborhood",
"stylers": [{"visibility": "off"}]},
                {"featureType": "road.arterial", "stylers":
[{"visibility": "off"}]},
                {"featureType": "road.highway", "elementType":
"labels", "stylers": [{"visibility": "off"}]},
                {"featureType": "road.local", "stylers":
[{"visibility": "off"}]}
            ]
        });

        $('#map').attr('data-zoom', map.getZoom());

        map.addListener('zoom_changed', function () {
            $('#map').attr('data-zoom', map.getZoom());

            if (z >= 8) {
                for (var i = 1; i <= 10; i++)
                    $('.turbine-pin').removeClass('x' + i);
                if ((z - 7) < 1)
                    z = 8;
                if ((z - 7) > 9)
```

```
                z = 16;
            $('.turbine-pin').addClass('x' + (z - 7));
        }
    });

    /**
     * Facilities
     */
    totalallPower = 0;
    var cntNoError = 0, spanAvgWindSpeed = 0, ActiveTurbinesWind
= 0, ActiveTurbinesGeo = 0;
    facilities.forEach(function (facility) {
        totalallPower += parseFloat(facility.power);

        var vStyleCentalMarker = '';
        var vStyleMarker = '';
        if (currentPlantId != '') {
            vStyleCentalMarker = 'el_hidden';
            if (facility.id == currentPlantId)
                vStyleMarker = 'el_shown';
        }

        if (facility.type == 'wind') {
            new CentralMarker(
                    facility,
                    facility.lat,
                    facility.lng,
                    map,
                    [
                        {id: facility.id},
                        {name: facility.name},
                        {lat: facility.lat},
                        {lng: facility.lng},
                        {power: facility.power},
                        {wind: facility.wind},
                        {clat: facility.center.lat},
                        {clng: facility.center.lng},
                        {type: facility.type}
                    ],
                    vStyleCentalMarker
                    );

            if (facility.buildings) {
                facility.buildings.forEach(function (turbine) {
                    new MarkerWind(
                            turbine.c,
```

```
                                turbine.d,
                                map,
                                [
                                    {type: turbine.b},
                                    {id: turbine.a},
                                    {lat: turbine.c},
                                    {lng: turbine.d},
                                    {power: turbine.e + ' kW'},
                                    {error: turbine.p},
                                    {plant: facility.id},
                                    {speed: turbine.g + ' m/s'},
                                    {error: turbine.p}
                                ],
                                'pin-item' + (vStyleMarker ? ' ' +
vStyleMarker : '')
                                );

                        //if (turbine.p == 0) {
                            spanAvgWindSpeed +=
parseFloat(turbine.g);
                            cntNoError++;
                        //}
                        if (turbine.e > 0) {
                            ActiveTurbinesWind++;
                        }
                    });
                }
                $('#imgCompass').css( 'animation-name', 'rotate' +
parseInt(facility.windDirection) );
            } else if (facility.type == 'geo') {
                new CentralMarker(
                        facility,
                        facility.lat,
                        facility.lng,
                        map,
                        [
                            {id: facility.id},
                            {name: facility.name},
                            {lat: facility.lat},
                            {lng: facility.lng},
                            {power: facility.power},
                            {clat: facility.center.lat},
                            {clng: facility.center.lng},
                            {type: facility.type}
                        ],
                        vStyleCentalMarker
```

```
                        );

                if (facility.buildings) {
                    facility.buildings.forEach(function (turbine) {
                        var info = turbine.e + ' KW'; // <br>' +
turbine.f + ' KW
                        new Marker(
                                turbine.c,
                                turbine.d,
                                map,
                                [
                                    {type: turbine.b},
                                    {id: turbine.a},
                                    {lat: turbine.c},
                                    {lng: turbine.d},
                                    {error: turbine.p},
                                    {plant: facility.id},
                                    {info: info},
                                    {energy: turbine.e},
                                    {reactiveenergy: turbine.f}
                                ],
                                'unit-marker' + (vStyleMarker ? ' ' +
vStyleMarker : '')
                                );
                        if (turbine.e > 0) {
                            ActiveTurbinesGeo++;
                        }
                    });
                }
            }

            if (facility.id == currentPlantId) {
                $('#spanCurPlant').html(facility.name + ' TOPLAM');
                $('#spantotalallPower').html(facility.power + ' MW');
            }
        });

        if (currentPlantId == '') {
            $('#spanCurPlant').html('TOPLAM');
            $('#spantotalallPower').html(totalallPower.toFixed(2) + '
MW');
        }

        $('#spanActiveTurbines').html('' + ActiveTurbinesWind + ' + '
+ ActiveTurbinesGeo);
        if (cntNoError > 0)
```

```
            spanAvgWindSpeed /= cntNoError;
        $('#spanAvgWindSpeed').html(spanAvgWindSpeed.toFixed(2) + '
m/s');
    }
}
```