Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації

Кафедра комп'ютерних систем управління

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Аналітична веб-система побутового споживання енергії

Виконав: студент 2 курсу, групи 2АКІТ-21м
спеціальності 151 – Автоматизація та комп'ютерно-інтегровані технології

Сергій Саранчук

Ім'я ПРІЗВИЩЕ

Керівник: д.т.н., професор, професор каф. КСУ

степінь, звання, посада

Володимир Дубовой

Ім'я ПРІЗВИЩЕ

«12» грудня 2022 р.

Опонент: к.т.н., доцент, доцент каф. АІІТ

степінь, звання, посада

Володимир Гармаш

Ім'я ПРІЗВИЩЕ

«14» грудня 2022 р.

Допущено до захисту

Зав. кафедри КСУ

В'ячеслав КОВТУН

«19» грудня 2022

Вінниця ВНТУ – 2022 рік

Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації

Кафедра комп'ютерних систем управління

Рівень вищої освіти другий (магістерський)

Галузь знань – 15 – Автоматизація та приладобудування

Спеціальність – 151 – Автоматизація та комп'ютерно-інтегровані технології

Освітньо - професійна програма – Інтелектуальні комп'ютерні системи

ЗАТВЕРДЖУЮ

Завідувач кафедри КСУ

В'ячеслав КОВТУН

"03" жовтня 2022 року

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Саранчуку Сергію Васильовичу

1. Тема роботи Аналітична веб-система побутового споживання енергії

керівник роботи д.т.н. проф Дубовой В. М.

затверджені наказом ВНТУ від "14" вересня 2022 року №203

2. Термін подання студентом роботи "12" грудня 2022 року

3. Вихідні дані до роботи: експлуатаційні дані з об'єкту дослідження,

сучасна програмна архітектура, мова програмування Python.

4. Зміст текстової частини: 1– огляд предметної області; 2 – аналіз можливих рішень; 3 – розробка і тестування програмного модуля для системи аналізу побутового споживання енергії;

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): архітектура системи; робочий процес системи; приклад вихідних даних; інтерфейс користувача розробленої системи; лістинг програмного забезпечення.

1. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	виконання прийняв
4	Небава М.І., пофесор кафедри ЕПВМ		

Дата видачі завдання "03" жовтня 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Огляд предметної області	04.10.22р	21.10.22р	
2	Розробка програмного забезпечення та експериментальні дослідження	22.10.22р	11.11.22р.	
3	Підготовка економічної частини	12.11.22р.	07.12.22р.	
4	Оформлення пояснювальної записки і графічного матеріалу	08.12.22р.	16.12.22р.	
5	Попередній захист роботи	17.12.22р.	17.12.22р.	
6	Остаточний захист роботи	22.12.22р.	22.12.22р.	

Студент (підпис) **Сергій Саранчук** (*Ім'я ПРІЗВИЩЕ*)

Керівник роботи (підпис) **Володимир Дубовой** (*Ім'я ПРІЗВИЩЕ*)

АНОТАЦІЯ

УДК 621.374.415

Саранчук С. В. аналітична веб-система побутового споживання енергії. Магістерська кваліфікаційна робота зі спеціальності 151 – Автоматизація та комп’ютерно-інтегровані технології, освітня програма – Інтелектуальні комп’ютерні системи. Вінниця: ВНТУ, 2022. 118 с.

У магістерській кваліфікаційній роботі розроблено аналітичну web систему побутового споживання енергії. Вона дозволяє покращити процес розрахунку вартості та подачі лічильників спожитої електроенергії. Попередньо провено аналіз існуючих програмних продуктів, які вирішують схожі проблеми. У загальній частині роботи розглянуто особливості побудови сучасних web додатків, а також обґрунтована доцільність їх розробки.

У економічній частині проаналізовано ринок подібних систем і розрахована планова собівартість продукції.

Ключові слова: електроенергія, тариф, показники.

ABSTRACT

УДК 621.374.415

Saranchuk S. V. analytical web system of household energy consumption. Master's thesis on specialty 151 - Automation and computer-integrated technologies, educational program - Intelligent computer systems. Vinnytsia: VNTU, 2022. 118 p.

An analytical web system for household energy consumption was developed in the master's thesis. It allows you to improve the process of calculating the cost and submitting meters of consumed electricity. An analysis of existing software products that solve similar problems has been previously conducted. In the general part of the work, the peculiarities of building modern web applications are considered, as well as the justified feasibility of their development.

In the economic part, the market of similar systems is analyzed and the planned production cost is calculated.

Keywords: electricity, tariff, indicators.

ВІДГУК

керівника магістерської кваліфікаційної роботи

студента Саранчука Сергія Васильовича

на тему «Аналітична веб система побутового споживання енергії»

Аналіз побутового споживання енергії є актуальною задачею в умовах дефіциту електричної і теплової енергії в Україні.

Студентом було проведено аналіз та порівняння можливих методів розв'язання поставленої задачі та обрано оптимальний варіант. Крім того, було досліджено існуючі реалізації розв'язання подібних задач. Розроблено веб-систему збору даних щодо побутового споживання енергії.

Дипломник показав хороший рівень спеціальних знань і «м'яких» навичок. Дипломник продемонстрував вміння: - вирішувати поставлені керівником завдання самостійно, згідно власноруч розробленої схеми заходів; - здійснювати пошук і узагальнення інформації; - комунікативні навички. Доведенням ерудиції та креативності дипломника є вчасно представлена магістерська кваліфікаційна робота.

Основні результати, представлені в роботі отримані дипломником самостійно. Матеріалу роботи властивий високий ступінь оригінальності, що доведено результатами перевірники на наявність запозичень.

Під час виконання магістерської кваліфікаційної роботи студент Саранчук С.В. проявив себе грамотним, кваліфікованим спеціалістом здатним приймати самостійно складні технічні рішення.

Недоліки роботи: не в повній мірі реалізована аналітична частина завдання.

Висновок

В представленій роботі на достатньому рівні вирішена науково-технічна задача, яка має практичне значення. Робота за змістом, актуальністю, новизною і практичною цінністю є завершеною науково-практичною роботою і відповідає вимогам до магістерської кваліфікаційної роботи. Автор магістерської роботи, Саранчук С.В. А., заслуговує на присудження кваліфікації магістра за спеціальністю 151 – Автоматизація та комп'ютерно інтегровані технології. Рекомендована оцінка випускної роботи «В».

Керівник магістерської кваліфікаційної роботи

Професор каф. КСУ ВНТУ
д.т.н., професор  Володимир ДУБОВОЙ

ВІДГУК

опонента магістерської роботи

студента Саранчука Сергія Васильовича

на тему «Аналітична веб система побутового споживання енергії»

Актуальність роботи в контексті спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» доведена результатами інформаційного пошуку та аналізу літературних джерел.

Перші 2 розділи магістерської кваліфікаційної роботи повністю присвячені огляду літературних та інформаційних джерел за обраною темою. Проаналізовано не менш ніж три аналоги створюваної системи. Функціональність та форм-фактор створеної системи цілком визначені на основі результатів критичного огляду літератури.

Прийняті рішення обґрунтовані результатами огляду літератури, результатами проектування та втілені в функціонуючу програмну систему. Результати її тестування доводять правильність прийнятих рішень.

Недоліки: Аналітична частина реалізована не в повній мірі. В тексті дипломної роботи зустрічаються поодинокі вади форматування, граматичні полики, семантичні неточності. Не всі рисунки достатньо якісні.

Загалом магістерська кваліфікаційна робота відповідає спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології», заслуговує на оцінку ____B____, а її автор заслуговує присудження кваліфікації: ступінь вищої освіти магістр, спеціальність «Автоматизація та комп'ютерно-інтегровані технології», освітня програма «Інтелектуальні системи і Інтернет речей».

Опонент

Доцент кафедри АІІТ ВНТУ Гармаш В.В.

ЗМІСТ

ВСТУП

Актуальність теми. В усьому світі соціально-економічні умови проживання населення значною мірою визначаються рівнем електроспоживання. Електрична енергія забезпечує комфортність у побуті (освітлення, тепло, телебачення тощо) і є ключовим чинником під час реалізації життєво необхідних умов існування населення (водопровід, центральне опалення, ліфти, електричний транспорт тощо).

Без електричної енергії неможлива робота промисловості та сфери обслуговування, які забезпечують зайнятість, оплату праці та оздоровлення населення. Стабільне, якісне постачання енергією населення та промисловості – невід'ємна умова економічного розвитку країни.

А для покращення стану та розвитку енергетичної системи України необхідним кроком є аналітика побутового споживання енергії. У сучасному світі найкраще це робити за допомогою аналітичних веб-систем побутового споживання енергії.

Метою роботи є створення аналітичної веб-системи побутового споживання електроенергії, як інструменту отримання інформації та аналітики електроенергії.

Завдання дослідження:

- Проаналізувати існуючі веб-системи побутового споживання електроенергії.
- Визначити кращі сервіси і взяти їх за зразок.
- Розробити макет (прототип) вебсайту та дизайн користувацького інтерфейсу, зробити верстку вебсайту по дизайну.
- Провести аналіз необхідних для створення вебсайту мов програмування, систем керування базами даних та фреймворків.
- Створити власну веб-систему побутового споживання енергії.

Об’єкт дослідження – процес аналізу побутових витрат енергії.

Предмет дослідження – програмне забезпечення власної аналітичної веб-системи побутового споживання енергії.

У роботі використовувались теоретичні **методи** та експериментальні дослідження, методи математичної статистики та статистичні вибірки. Також були використані методи прогнозування для розрахунку можливого зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки.

Новизна одержаних результатів полягає в розробці рекомендацій щодо впровадження веб-системи побутового споживання енергії.

Практичне значення. Розроблений веб-сайт дозволяє удосконалити сервіси подачі показів лічильника, тарифи за електроенергію, калькулятор розрахунку вартості електроенергії, подання заявки на підключення та зворотній зв'язок. Результати дослідження впроваджено в процес організації зберігання, опрацювання, передавання інформації шляхом створення та використання вебсайту побутового споживання енергії.

Структура і обсяг роботи. Магістерська робота на тему: "Аналітична веб-система побутового споживання енергії" складається зі вступу, 5 розділів, висновків, переліку використаних джерел та додатків. Загальний обсяг роботи складає 104 сторінки основного тексту, в тому числі 57 рисунків, 8 таблиць, список літератури з 50 найменувань та 2 додатків.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА АНАЛІЗ ІСНУЮЧИХ ВЕБ-СИСТЕМ ПОБУТОВОГО СПОЖИВАННЯ ЕНЕРГІЇ

Відповідно із завданням роботи необхідно розробити програмний продукт: аналітичну веб-систему побутового споживання енергії. Виконуючи поставлене завдання, буде виконанний аналіз аналогів (конкурентних методів) програмних продуктів, при цьому будуть відібрані за критеріями кращі практики, які ми використаємо для впровадження їх в нашу веб систему.

Для кращого дослідження та аналізу існуючих веб-систем розробимо критерії порівняння.

В рейтинговій таблиці ми будемо заповнювати поля в десятибальній системі, де 0 – найнижча відповідність критерію, а 10 – найвища. Заповнену таблицю можна переглянути в пункті «Підсумки огляду». Назви заголовків стовпців таблиці мають такі значення:

- № - порядковий номер веб-системи
- Назва – назва веб-системи
- Дизайн – оформлення зовнішнього вигляду веб-системи
- Функціональність – наявність зручного функціоналу введення, редагування, отримання даних
- Швидкість – швидкість відклику системи а дії користувача
- Сапорт – наявність у веб-системі зворотнього зв’язку.

1.1 Огляд аналогів програмних продуктів

Перш, ніж ми перейдемо до порівняння аналогів веб-систем побутового споживання енергії, розглянемо деякі теоретичні аспекти.

Зараз, і надалі під поняттям «енергія» ми впершу чергу будемо розуміти поняття «електроенергія», а хімічну, механічну, теплову та атомну енергії розглядати не будемо.

Електроенергія – це фізичний термін, що відображає здатність електричного струму виконувати механічну роботу, виділяти тепло чи випромінювати світло.

Чи заряджаєте ви свій смартфон або переглядаєте веб-сторінки, електрична енергія є невід'ємною частиною вашого повсякденного життя. Цей термін складається з двох компонентів - "електричний" та "енергія". Термін "енергія" може мати різні значення. У цій статті ви можете думати про неї як про потенційну енергію. За допомогою слова "електричний" вам дають поняття, що тут мають на увазі потенційну енергію електрично зарядженої частинки.

Подібно до того, як ваша потенційна енергія збільшується, коли ви піднімаєтеся в гору, електроенергія позитивної частки збільшується, коли вона “дереться” в електричному полі. Електричне поле залишає електричний потенціал у кожній точці простору (аналогічно гірському ландшафту, що має різну висоту у кожній точці). Під “підйомом вгору електричним полем” мається на увазі, що позитивна частина переміщається з точки з низьким електричним потенціалом в точку з вищим електричним потенціалом.

Оскільки електрична енергія одна із форм енергії, вона має одиницю виміру – джоуль, скорочено [Дж].

Щоб дати вам уявлення про те, скільки становить 1 Дж електричної енергії, ось невеликий приклад: для того, щоб світлодіодна лампа потужністю 1 Вт горіла протягом однієї секунди, вам потрібна електрична енергія в 1 Дж.

Ватт – це одиниця виміру потужності. Потужність P окреслюється робота за одиницю часу, тобто. $P = W / t$.

Знаючи, наведені вище теоретичні положення, виконаємо аналіз аналогів програмних продуктів.

1.1.1 Київська обласна ЕК

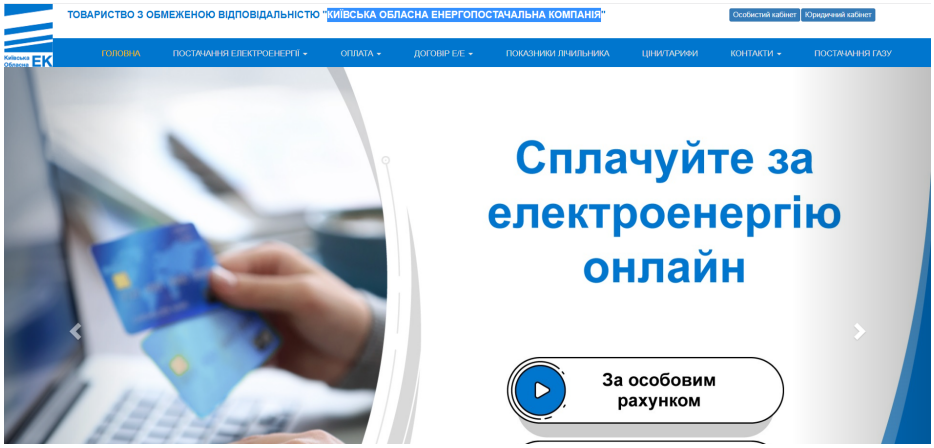


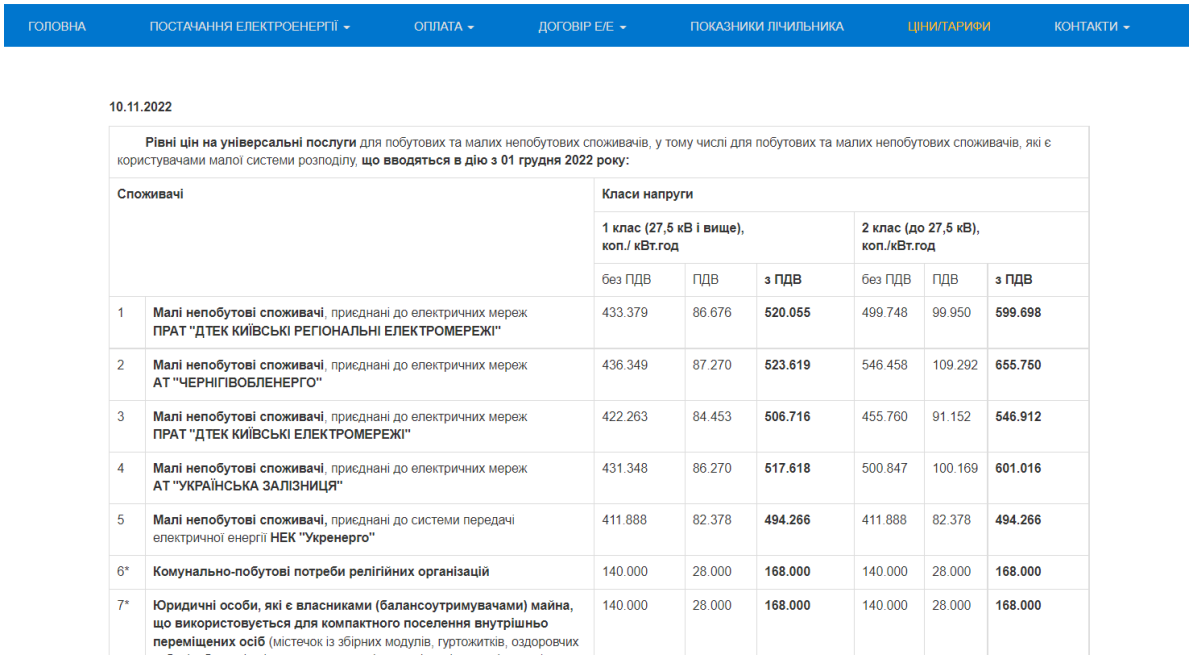
Рисунок 1.1 - Київська обласна ЕК – вигляд головної сторінки вебсайту

"Київська обласна енергопостачальна компанія" це компанія, яка надає можливість [1]:

- сплачувати рахунки за електроенергію без комісії
- передавати показники лічильника
- контролювати стан особового рахунку
- додавати до 5 особових рахунків
- дізнатися про діючі тарифи
- поширює інформацію про краще використання енергії протягом дня у зв’язку з військовими діями:

Рисунок 1.2 - Розумне споживання електроенергії

Після детального дослідження та аналізу веб-системи виявлено, що згідно нашим критеріям компанія має всі плюси у таблиці. Особливо вразила відкритість та своєчасність цін на тарифи для побутових та малих непобутових споживачів, які при зміні одразу оновлюються. Мною було вирішено взяти за зразок елементи такого виводу тарифів у своїй створюваній веб-системі.



ГОЛОВНА | ПОСТАЧАННЯ ЕЛЕКТРОЕНЕРГІЇ | ОПЛАТА | ДОГОВІР Е/Е | ПОКАЗНИКИ ЛІЧИЛЬНИКА | ЦІНИ/ТАРИФИ | КОНТАКТИ

10.11.2022

Рівні цін на універсальні послуги для побутових та малих непобутових споживачів, у тому числі для побутових та малих непобутових споживачів, які є користувачами малої системи розподілу, **що вводяться в дію з 01 грудня 2022 року:**

	Споживачі	**Класи напруги**					
		1 клас (27,5 кВ і вище), коп./ кВт.год			**2 клас (до 27,5 кВ), коп./кВт.год**		
		без ПДВ	ПДВ	**з ПДВ**	без ПДВ	ПДВ	**з ПДВ**
1	**Малі непобутові споживачі**, приєднані до електричних мереж **ПРАТ "ДТЕК КИЇВСЬКІ РЕГІОНАЛЬНІ ЕЛЕКТРОМЕРЕЖІ"**	433.379	86.676	**520.055**	499.748	99.950	**599.698**
2	**Малі непобутові споживачі**, приєднані до електричних мереж **АТ "ЧЕРНІГІВОБЛЕНЕРГО"**	436.349	87.270	**523.619**	546.458	109.292	**655.750**
3	**Малі непобутові споживачі**, приєднані до електричних мереж **ПРАТ "ДТЕК КИЇВСЬКІ ЕЛЕКТРОМЕРЕЖІ"**	422.263	84.453	**506.716**	455.760	91.152	**546.912**
4	**Малі непобутові споживачі**, приєднані до електричних мереж **АТ "УКРАЇНСЬКА ЗАЛІЗНИЦЯ"**	431.348	86.270	**517.618**	500.847	100.169	**601.016**
5	**Малі непобутові споживачі,** приєднані до системи передачі електричної енергії **НЕК "Укренерго"**	411.888	82.378	**494.266**	411.888	82.378	**494.266**
6*	**Комунально-побутові потреби релігійних організацій**	140.000	28.000	**168.000**	140.000	28.000	**168.000**
7*	**Юридичні особи, які є власниками (балансоутримувачами) майна, що використовується для компактного поселення внутрішньо переміщених осіб** (містечок із збірних модулів, гуртожитків, оздоровчих	140.000	28.000	**168.000**	140.000	28.000	**168.000**

Рисунок 1.3 - Ціни на тарифи для побутових споживачів

Також високої оцінки заслуговує support, зворотній зв’язок компанії. Окрім стандарної інформації «Про компанію» та «Режим роботи» є й посилання на «КОЛ-ЦЕНТР», «СТОП КОРУПЦІЯ. Телефон довіри» та «Оператори систем розподілу».

1.1.2 АТ "Полтаваобленерго"

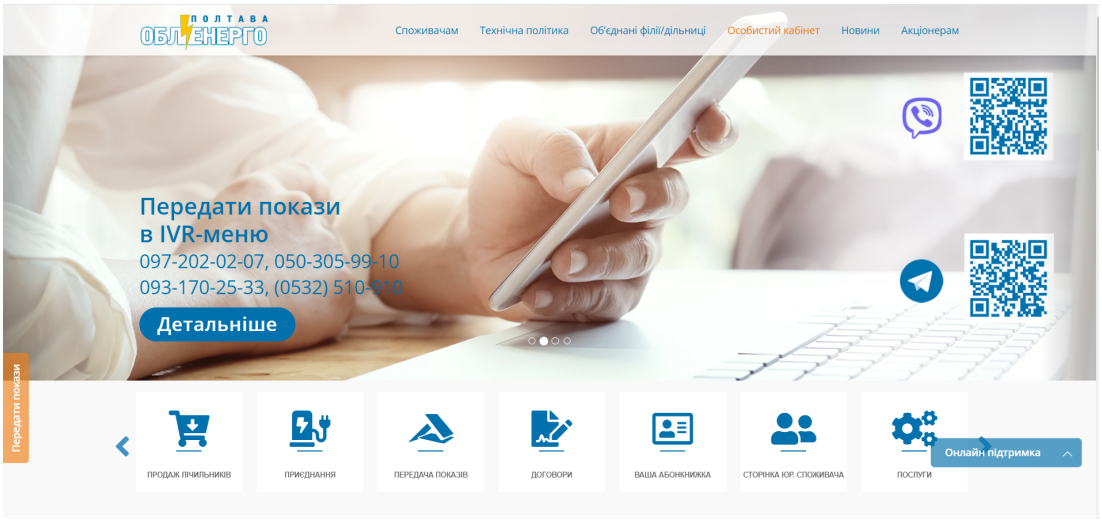


Рисунок 1.4 - Полтаваобленерго – вигляд головної сторінки вебсайту

Акціонерне товариство Полтаваобленерго[2] є спеціалізованою мережевою організацією, основний вид діяльності якої – надання послуг із передачі електричної енергії за допомогою здійснення комплексу організаційно та технологічно пов'язаних дій, що забезпечують передачу електричної енергії через технічні пристрої.

Лише тільки передати показники можна сімома способами:

- через сайт АТ «ПОЛТАВАОБЛЕНЕРГО» без реєстрації, у меню «Передати покази»;
- на e-mail pokaz@poe.pl.ua;
- через онлайн-сервіс «Ваша абонентська книжка»;
- через чат-бот «ПолтаваОблЕнерго»
- через мобільний додаток, перейшовши за посиланням;
- через месенджер Viber;
- у месенджері Telegram @PltOblEnergo_bot;

У онлайн-чаті можна поділитися своєю думкою про роботу, позначити проблеми, які потребують вирішення, отримати відповіді на питання, які стосуються діяльності організації.

Інформація про персональні дані громадян і компаній, що направили запит в електронному вигляді, зберігається та обробляється з дотриманням вимог українського законодавства про персональні дані і не публікується у відкритому доступі.

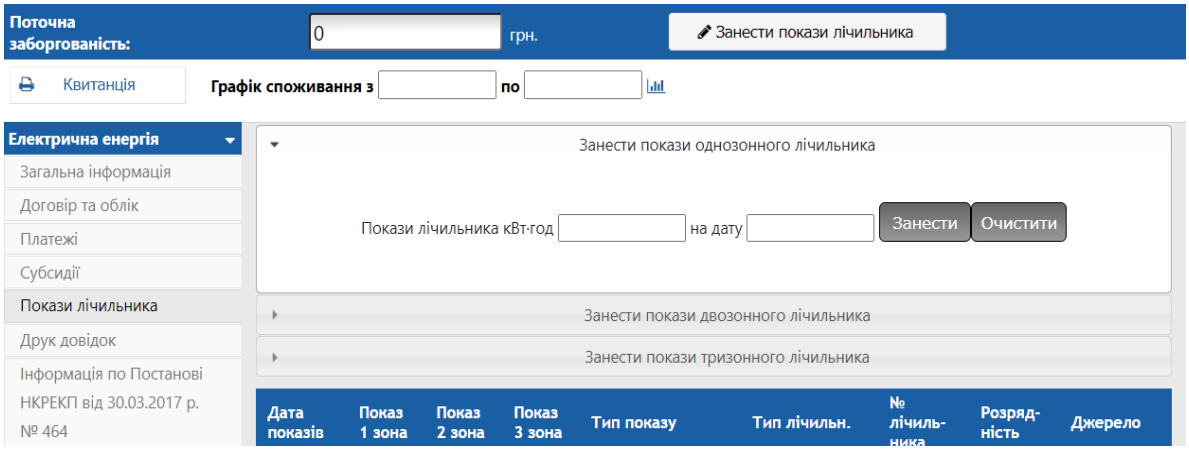


Рисунок 1.5 - Внесення показів лічильника в особистому кабінеті

Великим плюсом є велика функціональність особистого кабінету, дуже зручний спосіб передачі показників лічильників. Візьмемо його за зразок в нашу веб-систему побутового споживання енергії.

Значним мінусом є мала швидкість завантаження данних. При перому запуску веб-сайта необхідно чекати декілька секунд. Вносимо всі результати дослідження в таблицю з критеріями.

Аналізуючи даний веб-ресурс приходиш до думки, що на цьому сайті є вся необхідна для побутового споживача інформація, але сайт створено так, що якщо користуватися ним щодня, тоді буде працювати з ним дуже зручно, а якщо заходити на нього раз на місяць, або ще рідше, тоді пошук інформації на ньому стає менш зручним, тому це вплинуло на оцінку критерію «Зручність».

1.1.3 Калькулятор розрахунку оплати за електроенергію

Витрата електроенергії в кожній сім'ї залежить від потужності використовуваних приладів та часу їх дії. За електроенергію необхідно платити за встановленим тарифом. Щоб самостійно визначити кількість спожитої за

місяць електроенергії або виконану струмом роботу, необхідно: визначити показання лічильника на початку та наприкінці місяця; знайти різницю показань лічильника на початку та наприкінці місяця (кількість спожитої електроенергії протягом місяця в кіловат-годинах); отриману кількість електроенергії помножити на тариф.

Споживання електроенергії в кіловат-годинах показують лічильники електроенергії. Вони бувають різними на вигляд і типу конструкції.

Рисунок 1.6 - Види лічильників електроенергії

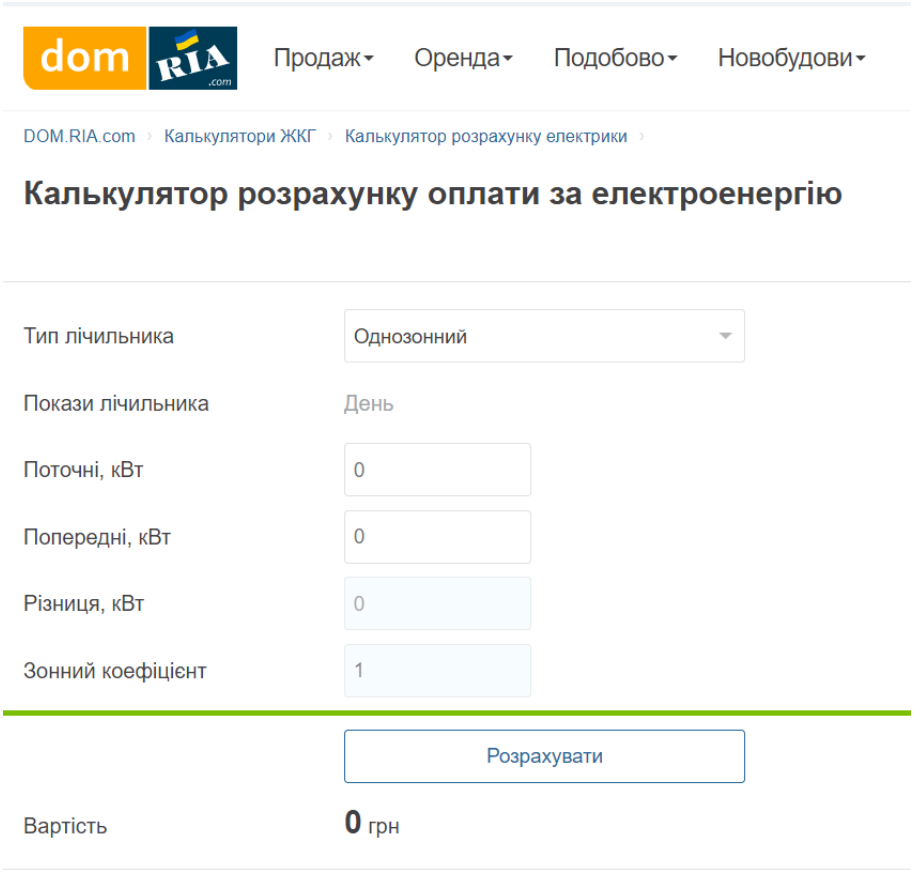


Рисунок 1.7 - Калькулятор розрахунку оплати за електроенергію

На веб-ресурсі розрахунку оплати за електроенергію розрахунок електроенергії за лічильником проводиться шляхом множення спожитої кількості електроенергії (кВт · год) на тариф (грн / кВт · год).

До спожитої електроенергії до 100 кВт·год, застосовується тариф 0,9 грн/кВт·год, а від 100кВт·год – 1,68 грн/кВт·год. Таким чином, перші 100 кВт·год обчислюємо за формулою:

кВт·год × 0,9 грн/кВт·год,

а решта – кВт·год × 1,68 грн/кВт·год

Для двухзонного лічильника розрахунок проводиться окремо для денного і нічного часу. У нічний час (з 23.00 до 7.00) тариф розраховується зі зниженим коефіцієнтом 0,5, а в денний (з 7.00 до 23.00) - з коефіцієнтом 1. Щоб розрахувати електроенергію по двозонному лічильнику, спочатку потрібно визначити частку електроенергії, спожитої вдень і вночі.

Для дня: кВт·год(день)/(кВт·год(день) + кВт·год(ніч))

Для ночі: кВт·год(ніч)/(кВт·год(день)+кВт·год(день))

Для трьохзонного лічильника виділяють періоди: пік – (з 8.00 до 11.00 і з 20.00 до 22.00) з коефіцієнтом 1,5; напівпік – (з 7.00 до 8.00, з 11.00 до 20.00 і з 22.00 до 23.00) – коефіцієнт 1; ніч – (з 23.00 до 7.00) – коефіцієнт 0,4. Спочатку обчислюємо частку спожитої електроенергії в піковий, напівпіковий і нічний період.

Для пікового періоду:
кВт·год(пік)/(кВт·год(пік)+кВт·год(напівпік)+кВт·год(ніч))

Для напівпікових періоду:
кВт·год(напівпік)/(кВт·год(пік)+кВт·год(напівпік)+кВт·год(ніч))

Для ночі: кВт·год(ніч)/(кВт·год(пік)+кВт·год(напівпік)+кВт·год(ніч)) [3]

Розраховуючи вартість обирається:

- Тип лічильника (однозонний, двозонний, трьохзонний)
- Поточні, кВт
- Попередні, кВт

Після цього автоматично розраховується:

- Різниця, кВт
- Зонний коефіцієнт

Натискаючи на кнопку «Розрахувати» ми отримуємо «Вартість».

На цьому ресурсі дуже високі показники швидкісті та зрузночі, тому будемо використовувати ці значення та формули при додаванні розрахунку енергії на створюваний нами веб-ресурс.

1.1.4 Діючі тарифи для населення ТОВ – Полтаваенергозбут

Полтаваенергозбут показує завжди актуальну інформацію щодо цін на електричну енергію [4]. Дуже вдалий дизайн в поєднанні з ергономікою об’єктів. Лише з головної сторінки сайту можна побачити всі основні розділи ресурсу, а також діючі тарифи та архів з неактуальними тарифами.

Також є фахівці інформаційно-консультаційного центру надають консультації з питань:

- застосування чинного законодавства України у сфері енергетики;
- прав і обов’язків електропостачальника та споживача;
- роботи структурних підрозділів ТОВ «ПОЛТАВАЕНЕРГОЗБУТ».
- При ІКЦ діє незалежна комісія з розгляду і розв’язання суперечностей між електропостачальником та споживачами.

Рисунок 1.8 Полтаваенергозбут головна сторінка веб-ресурсу

Великий інтерес викликає викликають підсторінки сторінки «Споживачам» де в п'яти пунктах надається вся необхідна інформація споживачам. Перелік цих пунктів:

- Споживачам універсальної послуги
- Споживачам ринку вільних цін
- Побутовим споживачам
- Нормативні документи
- Корисна інформація.

Візьмемо деякі з цих пунктів за основу для нашого веб-ресурсу.

1.1.5 ПрАТ «Закарпаттяобленерго»

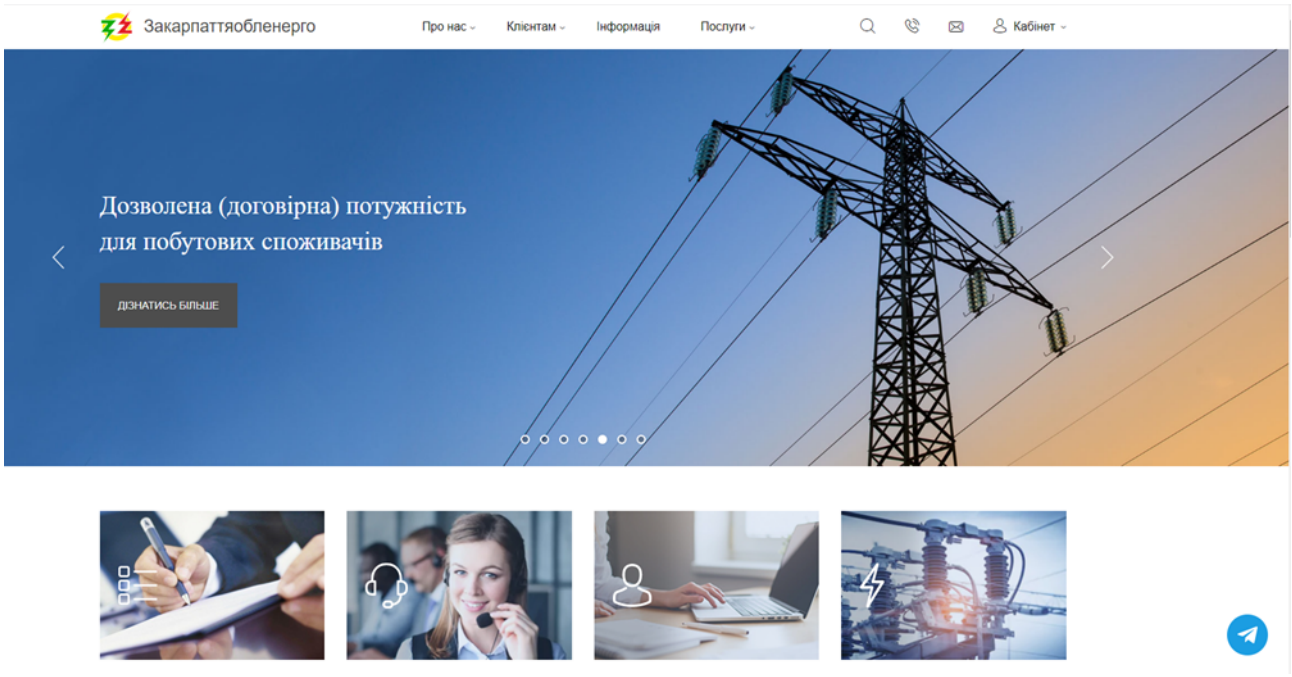


Рисунок 1.9 Закарпаттяобленерго – вигляд головної сторінки вебсайту

Клієнтам надаються наступні послуги:

- Тарифи
- Договір споживача про надання послуг з розподілу
- Учасникам ринку електроенергії
- Інформація для населення
- Інформація для юридичних осіб
- Перерви у електропостачанні
- Інформаційно-консультаційний центр
- Нормативні документи
- Прийом громадян

Особисто для мене дизайн цього веб-ресурсу дуже вдалий та зручний, тому і отримує найвищі оцінки одноіменних впроваджених критеріїв і елементи будуть використанні як зразок для нашого веб-ресурсу.

Сайт має велику базу інформації для споживачів:

- Питання та відповіді стосовно державної реформи ринку електричної енергії
- Цільові програми
- Річна, квартальна та особлива інформація
- Інформація товариства, що підлягає оприлюдненню
- Геодезичні інформаційно-технічні дані об'єктів електромереж
- Охорона праці

За функціональністю наданих послуг веб-ресурс трохи поступається Київській обласній ЕК та Полтаваобленерго, також судячи з відгуків зворотній зв’язок від компанії бажає кращого [6].

Підсумки огляду

Дослідивши зазначені веб-сервіси, які представлені в пунктах 1.1.1 – 1.1.5, робимо висновок, що всі вони є аналітичними веб-системами побутового споживання енергії . Кожен веб-сайт дозволяє отримувати актуальну інформацію щодо тарифів, внесення показників, новин, розрахунку вартості електроенергії.

Певні з досліджених веб-ресурсів мають всі зазначені категорії, а інші лише деякі з них.

У результаті аналізу даних сервісів робимо висновок: серед наведених прикладів не існує програмного продукту, що володів би повним набором необхідних функцій з максимальною оцінкою, тому ми спробуємо вдосконалити це у нашому створюваному веб-сервісі. Отримані дані з огляду розроблених програмних продуктів були зведені в таблицю 1.1.

Таблиця 1.1 - Аналіз розроблених програмних продуктів

№	Назва	Зручність	Дизайн	Функціо-нальність	Швидкість	Support
1	ТОВ «Київська обласна ЕК»	8	7	10	9	10
2	АТ "Полтаваобленерго"	7	9	10	5	9
3	Калькулятор розрахунку оплати за електроенергію	10	8	9	10	8
4	ТОВ «Полтаваенергозбут»	9	10	9	9	9
5	ПрАТ «Закарпаттяобленерго»	10	10	9	7	6

Проаналізувавши всі дані таблиці бачимо що критерій «Дизайн» ми візьмемо за зразок у ТОВ «Полтаваенергозбут» та ПрАТ «Закарпаттяобленерго», критерій «Функціональність» у ТОВ «Київська обласна ЕК» та АТ "Полтаваобленерго", критерій «Support» у ТОВ «Київська обласна ЕК», критерій «Швидкість» та «Зручність» будемо рівняти на «Калькулятор розрахунку оплати за електроенергію», і також, зручність повинна бути як у ПрАТ «Закарпаттяобленерго».

РОЗДІЛ 2. АНАЛІЗ МОЖЛИВИХ РІШЕНЬ

Веброзробка – процедура створення вебдодатка або вебсайту. Основними етапами цього процесу є такі заходи, як дизайн, верстка сторінок сайту, вебпрограмування на стороні сервера та клієнта, а також роботи зі конфігурування вебсервера.

Так як існує багато технологій, які можна використовувати для розробки веб-систем, необхідно провести аналіз можливих рішень, щоб вибрати ті технології, які найкраще підійдуть для вирішення наших завдань.

Основні аспекти, на які слід звернути увагу і провести їх порівняльний аналіз, це:

- мова програмування та платформа для розробки (IDE);
- системи керування базами даних;
- фреймворки, які прискорють розробку.

2.1 Вибір мов програмування

Як показує статистика Stack Overflow — популярної системи для професійних програмістів [7], найпопулярнішою мовою програмування є Javascript [10], далі об’єднано йдуть мова гупертекстової розмітки HTML [19] та каскадні таблиці стилів CSS [21]. На третьому місці мова програмування Python [12] та четверта у списку мова структурованих запитів SQL [22]. Забігаючи наперед скажу, що саме ці технології ми використаємо для створення аналітичної веб-системи побутового споживання енергії.

HTML/CSS та JavaScript (JS) це основний стек технологій для створення фронтенду (Front-End) – зовнішнього вигляду сайту. Цей набір використовується на всіх без виключення сучасних вебсайтах. Якщо раніше, перші вебсторінки писалися лише на HTML, то згодом функціоналу стало не вистачати і додалися каскадні таблиці стилів, ще через певний час «обов’язковим» став JS. Розглянемо детальніше вибрані мови.

Мову програмування **JavaScript** розробили спеціально для того, щоб створювати інтерактивні сайти.

Код мовою JavaScript називають скриптом. Його зберігають в окремий файл із розширенням js, а щоб запустити, підключають цей файл на сторінку. У HTML для додавання JavaScript є спеціальний тег:

<script src="адреса_файлу"></script>

Підключають скрипт зазвичай наприкінці сторінки, перед закриваючим тегом </body>.

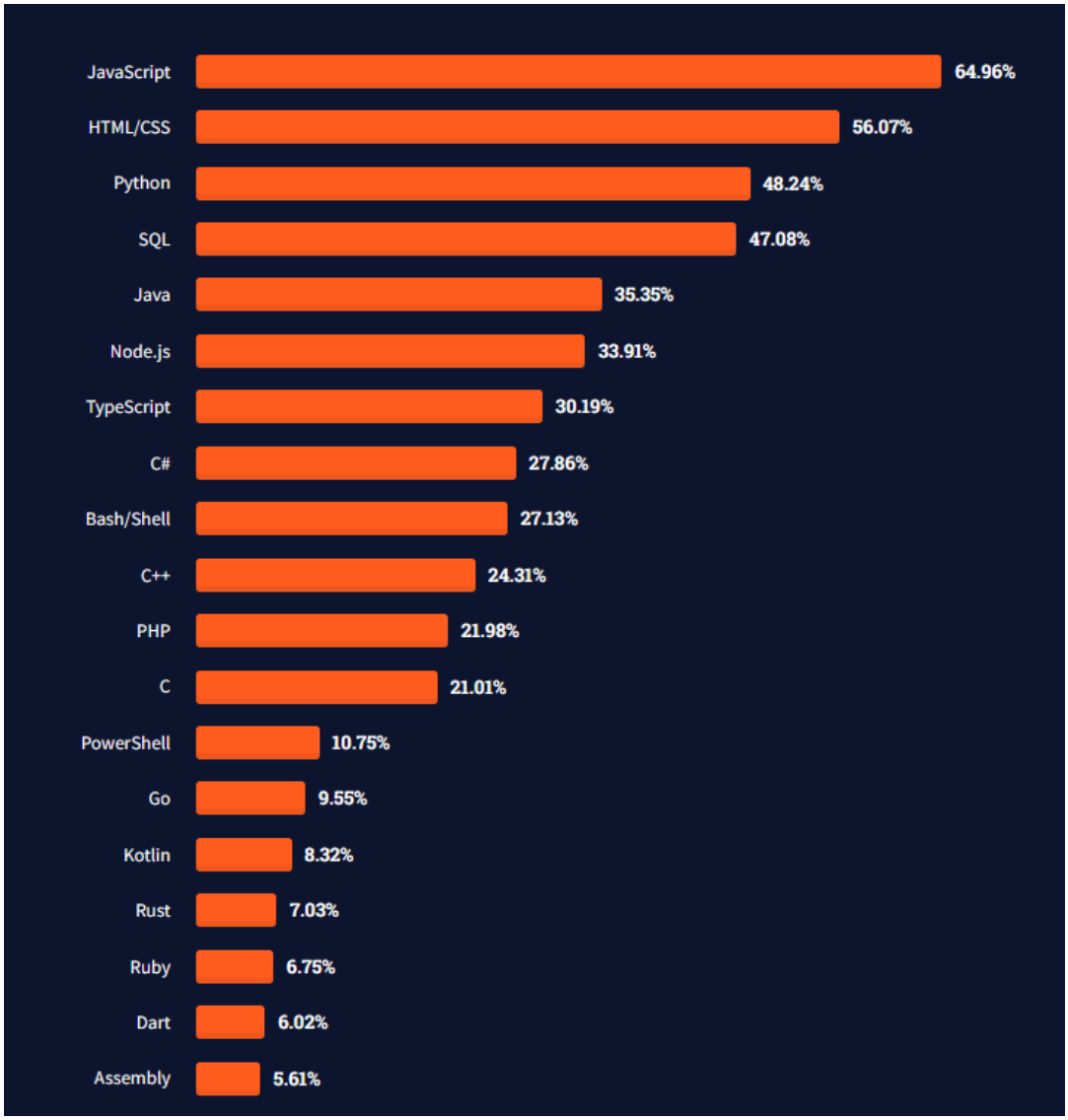


Рисунок 2.1 - Рейтинг мов програмування на сайті Stack Overflow

Так як Javascript – це клієнтська мова програмування, то це означає, що вона працює на стороні клієнта. Коли ми переходимо на будь-який сайт в Інтернеті, ми робимо це за протоколом HTTP. Зі свого домашнього комп'ютера ми надсилаємо запит на віддалений сервер, на якому розташовується сайт. І

віддалений сервер нам вже надсилає відповідь (html-сторінку, яка буде відображена на домашньому комп'ютері у браузері).

Клієнтом для протоколу HTTP є браузер. Це може бути Google Chrome, Firefox, Opera та інші.

Можна спрощено сказати, що мова програмування Javascript вбудована у можливості браузера. У разі встановлення браузера ви вже маєте можливості для роботи з мовою Javascript.

Звідси найголовніший плюс мови Javascript - це те, що для того, щоб ним користуватися не потрібно встановлювати будь-яке додаткове програмне забезпечення. Все вбудоване в браузер і маючи його, вже можна працювати з Javascript.

Головне завдання JavaScript – внести можливості автоматизації, на вебсторінки сайтів.

Насамперед ми будемо створювати інтерактивні елементи, з якими користувач може взаємодіяти. Можна взаємодіяти з цими елементами та отримувати якийсь оброблений результат [13].

Крім того, за допомогою Javascript і її бібліотеки JQuery [14] можна робити різні слайдери, каруселі, картинки, які змінюються самі собою.

Програма JavaScript — це послідовність інструкцій, тобто вказівок браузеру виконати якісь дії. Інструкції виконуються послідовно, зверху донизу.

Щоб сказати JavaScript, що інструкція закінчена, потрібно поставити крапку з комою або перейти на новий рядок. Новий рядок правильно працює в більшості випадків, а крапка з комою завжди. Тому краще ставити крапку з комою наприкінці кожної інструкції.

JavaScript не змінює вихідний файл із розміткою, але, виконуючи інструкції, змінює сторінку прямо у браузері користувача.

Консоль – інструмент розробника, що допомагає тестувати код. Якщо під час виконання скрипта виникне помилка, у консолі з'явиться повідомлення про неї. А ще у консоль можна виводити текстові підказки. Щоб вивести повідомлення до консолі, потрібно використовувати console.log.

JavaScript стежить за тим, що відбувається на сторінці. Клік по кнопці або відправлення форми – це подія. Ми можемо сказати JavaScript, що зробити, коли подія відбудеться. Для цього використовують обробники подій. Інструкції, які повинні будуть виконуватися, коли подія відбудеться, розташовують між фігурними дужками.

Наприклад властивість onclick означає «по кліку»:

```
let button = document.querySelector('button');
button.onclick = function() {
  console.log('Кнопка натиснута!');
};
```

Наступна в нашому списку **HTML** – це мова розмітки гіпертекстових документів. Вона потрібна, щоб відображати у браузері спеціальним чином відформатований документ з безліччю вкладених елементів: заголовками, абзацами, списками, гіперпосиланнями, медіаджерелами, розташуванням зображень, відео та аудіо.

HTML-документ це текстовий файл із розширенням .html або .htm. У браузері він перетворюється на веб-сторінку і складається з набору тегів. Вони таки допомагають представляти текст на екрані: завдяки їм браузер розуміє, що він читає не просто текст, а структуровану інформацію, розбиту на блоки.

Тег виглядає як набір символів, укладений у кутові дужки. Символи у дужках позначають ім'я тега, яке описує його функції. Ось кілька прикладів:

<h1> </h1> - заголовок;

*** - абзац;***

*** - курсив.***

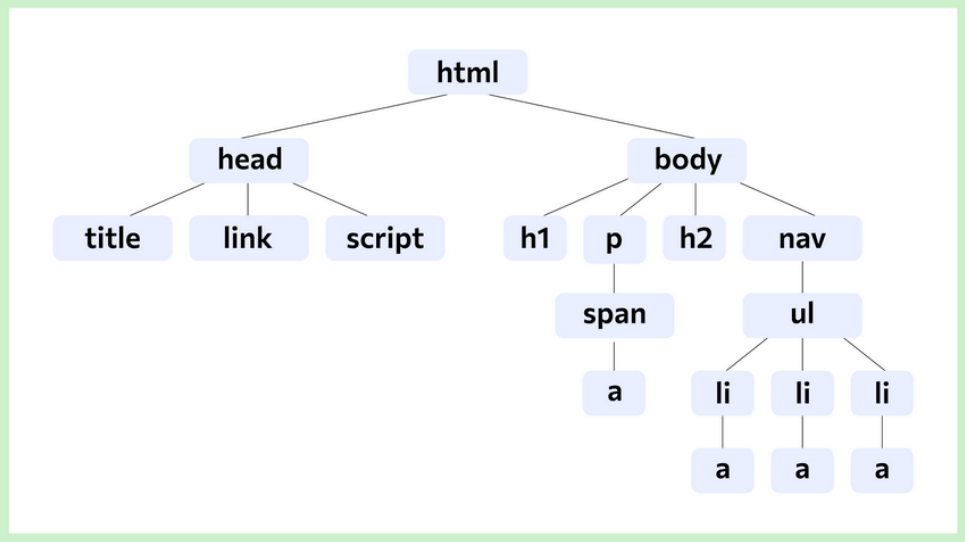


Рисунок 2.2 - Приклад структури HTML-документа

Тег - це складовий елемент, що визначає розмітку структурних блоків. Він відкривається, і це починає свою дію; та закривається, позначаючи завершення команди. Закриті та відкриті теги розрізняються лише слішем перед ім'ям тега. Ці теги створюють оболонку, в яку міститься текст.

Саме незакриті теги призводять до частих помилок та некоректних відображень сторінки. Для наочності уявімо, що теги - це матрьошки, з яких можна зібрати набір. Складаючи у велику матрьошку всі фігурки важливо не забувати закривати всі половинки (ставити теги, що закривають), інакше іграшка не вийде.

Всередині тега можуть бути атрибути – додаткова інформація, яку потрібно приховати з тексту. Вони ставляться тільки в тег, що відкриває, між ним і ім'ям тега повинен бути пробіл, а після нього йде знак рівності. Значення атрибуту полягає у лапки. За допомогою їх можна розширити можливості тегів і звернутися до них, щоб дізнатися детальну інформацію.

Є теги, які не потрібно закривати. Приклад: тег перенесення рядка
 — одиночний і закривати його не потрібно. Раніше одиночні теги писалися з закриваючим слешем перед дужкою. Наприклад:
. У стандарті HTML5 використання закриває слеша в одиночних тегах необов'язково. Приклади одиночних тегів:
, <hr>, <img>.

Крім атрибутів у тег можна додавати вкладення, ці елементи можуть змінювати стиль тексту. Наприклад, можна виділити слово **напівжирним** шрифтом.

Далі розглянемо **CSS** – це мова опису зовнішнього вигляду документа, тобто вона відповідає за те, як виглядають веб-сторінки: колір фону та декоративних елементів, розмір та стиль шрифтів. Термін розшифровується як Cascading Style Sheets (каскадні таблиці стилів). CSS взаємодіє з іншою мовою розмітки - HTML, яка відповідає за розміщення елементів на сторінці.

CSS, як і будь-яку мову, має синтаксис. У ньому є правила - значення, що визначають зовнішній вигляд елементів. CSS-правило складається з селектора, CSS-властивостей та їх значень:

Селектори — це мітки, які допомагають браузеру зрозуміти, до якої частини HTML коду потрібно застосувати задані параметри.

CSS-властивості — це певні параметри оформлення, наприклад, колір елемента або тексту (color) або колір фону (background).

Значення це просто значення, воно виражається текстом або числом, наприклад чорний (black).

селектор {	***p {***
властивість: значення;	***color: black***
}	***}***

CSS-правила в коді розташовуються у фігурних дужках {…}. Перед відкриттям дужки обов'язково потрібно вказати селектор, до якого належить це правило.

У прикладі селектор є <p>, і він вибирає всі теги з ім'ям <p>, color — це CSS-властивість a black — значення CSS-властивості. Зв'язка "властивість: значення" називається блоком оголошення стилів. Всередині нього властивість відокремлюється від значення двокрапкою, а один блок від іншого відокремлює крапка з комою.

Таблиці називаються каскадними, тому що працюють за принципом каскаду — тобто правило, прописане нижче, вважається пріоритетним.

Наприклад, якщо в прикладі під значенням фонового кольору ми пропишемо ще одне значення color: red, то колір тексту буде червоним, а не чорним.

```
p {
color: black
background: #eeeeee
color: red
}
```

Отже, із мовами для написання фронтенду ми визначилися це будуть HTML/CSS та Javascript, залишається вибрати одну із серверних мов для написання бекенду (Back-End). Основними серверними мовами для веб-сервісів є Python та PHP, нам необхідно порівняти та обрати одну із них для нашої вебсистеми.

PHP – від англійського Hypertext Preprocessor – «препроцесор гіпертексту». Це скриптова мова, нею створюють сайти та веб-додатки. Мова інтегрується з більшістю веб-серверів і працює з усіма найпоширенішими операційними системами. PHP має зрозумілий синтаксис і низький поріг входу для вивчення.

Плюси PHP:

- Висока продуктивність. Для веб-розробки швидкість виконання програм є ключовим параметром. У цьому PHP обходить більшість мов, включаючи Python.
- Робота із різними платформами. У веб-розробці важливо забезпечити стабільну якість, незалежно від рішення на стороні сервера операційної системи або веб-сервера. PHP підтримує Oracle, MySQL, Apache, Windows, Unix, Linux та інші платформи.
- Популярність. 79% сайтів написано на PHP, цю мову використовують поширені системи управління сайтами, наприклад WordPress, Drupal, Magento. Мова часто потрібна у вакансіях програмістів.

- Велике ком'юніті. Спільнота PHP-розробників більша, ніж у Python, тому і вибір бібліотек в області веб-розробки ширший. У ком'юніті простіше знайти потрібне керівництво чи отримати відповідь на запитання.

Мінуси PHP:

- Несистемний синтаксис. Наприклад, назви функцій може бути схожі, але виконувати різні операції. Також, коли мова допрацьовували, використовували мови C і Java, тому можна зустріти синтаксис. Розробник-початківець може заплутатися, а досвідчений, навпаки, побачить у цьому перевагу, тому що йому буде легше перейти на нові мови.
- Негативна слава. PHP створювалася як мова, яку може використовувати людина без підготовки. Тому його поверхньо освоювали недосвідчені програмісти та кидалися виконувати замовлення. Сайти ламалися, а горе-фахівці нічого не могли вдіяти. Хоча мова вдосконалюється і сучасний PHP нічим не гірший за молоді мови, все ще можна почути негативні думки.
- Можливість помилок. Якщо в коді буде помилка, то мова дозволить використовувати її. Коли недолік буде очевидним, помилку буде складно знайти. Тому, хоча PHP — проста і гнучка мова, неуважний розробник може створити проблеми собі та колегам.

Python – це об'єктно-орієнтована мова загального призначення, її застосовують у різних галузях. Наприклад, на ній пишуть програми, програмують системи машинного навчання, аналізують дані. Розробникам вона подобається за вбудовані структури даних, зручні функції кодингу та динамічні посилання. Це допомагає писати код на Python швидко і знижує ймовірність помилок [26].

Плюси Python:

- Простий код. Програми на Python [20] легко писати та читати. Якщо розробнику передадуть чужий код, він не витрачає багато часу, щоб у ньому розібратися. Також у простому синтаксисі легше знайти помилки та вразливості.

- Мінімум зайвих завдань. Розробнику не потрібно думати про технічні моменти, пов'язані з пам'яттю. Наприклад, Python автоматично видаляє об'єкти, до яких немає доступу.
- Кросфункціональність. Підтримка Python вбудована в різні програмні платформи та операційні системи, мову можна інтегрувати з Java, C та C++. Тому написаний на Python додаток не потрібно щоразу переписувати.
- Широкі можливості. Для Python існує багато різних бібліотек наборів готових функцій. Замість писати код з нуля, їх можна розгорнути буквально двома рядками коду та користуватися. Наприклад, маніпулювати математичними операціями, будувати нейромережі, автоматизувати процеси.

Мінуси Python:

- Низька швидкість. Операції на мові Python не можуть виконуватися паралельно, тому працюють повільніше та потребують більше пам'яті пристрою. Проте код на Python пишеться швидше, і буває, що це важливіше для замовника. Продуктивність можна підняти, наприклад, переписавши код іншою мовою і зв'язавши його з Python.
- Динамічна типізація. У Python можна створити змінну, не позначаючи якого типу даних до неї повинні входити — числа, текст або щось інше. Тоді розробник може думати, що складає кількість товарів, а насправді за змінною ховаються артикули. У результаті програма може бути написана правильно, але не працювати.

Порівнявши плюси та мінуси PHP та Python зупиняємо свій вибір на Python, як мову яка має більш простий сисинтаксис у порівнянні із PHP.

Ще із рейтингу обираємо мову SQL яку ми детально розглянемо в наступному пункті.

2.2 Вибір системи керування базами даних

Після обрання мов програмування наступним етапом є вибір системи керування базами даних (СКБД), часто можна зустріти назву системи управління базами даних (СУБД) – це ідентичні поняття.

Системи керування базами даних являють собою набір програмних та апаратних засобів, за допомогою яких можна проектувати, налаштовувати та адмініструвати бази даних (БД). СКБД гарантують цілісність, безпеку зберігання даних і дозволяють видавати доступ до адміністрування БД.

Для визначення найпопулярніших СКБД звернемося знову до сайта Stack Overflow і проаналізуємо топ 5 систем.

MySQL працює на Linux, Windows, OSX, FreeBSD та Solaris. Можна розпочати роботу з безкоштовним сервером, а потім перейти на комерційну версію. Ліцензія GPL з відкритим вихідним кодом дозволяє модифікувати програмне забезпечення MySQL.

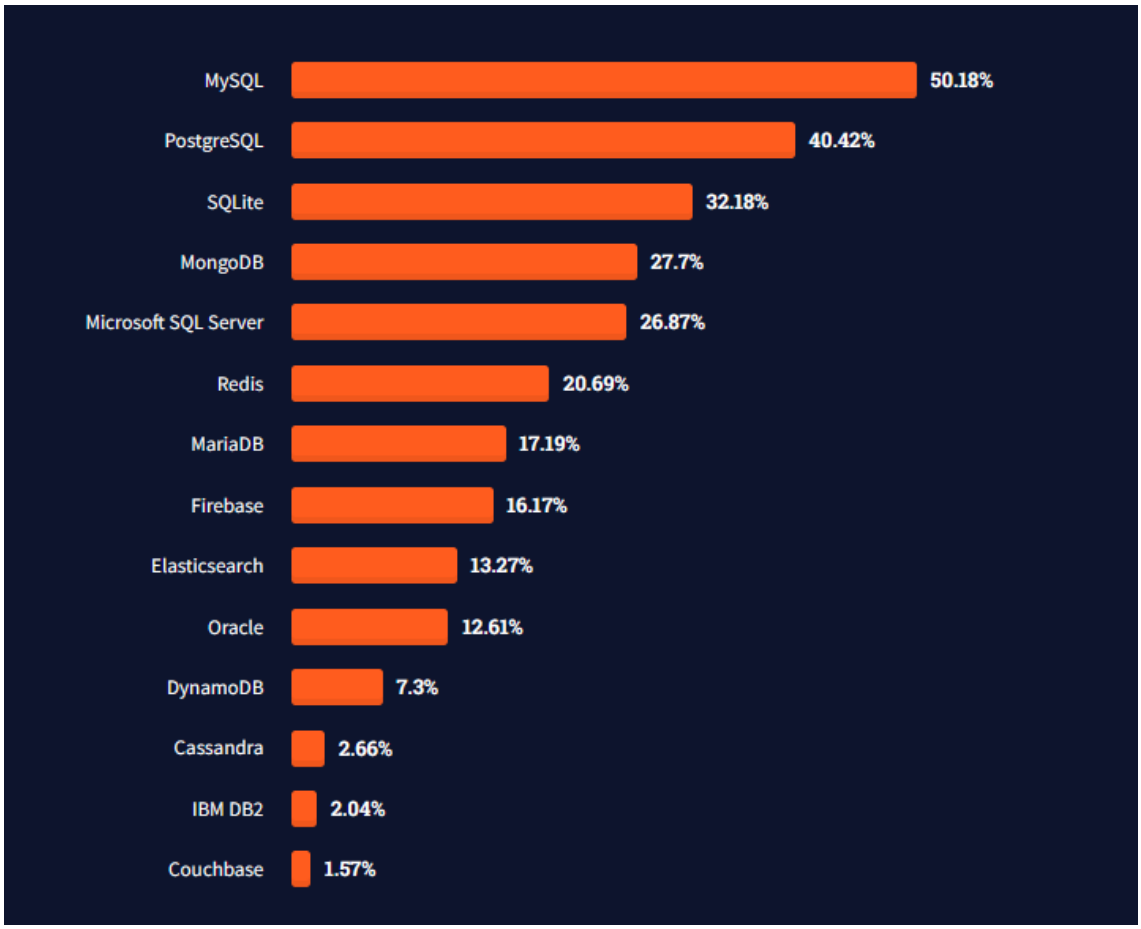


Рисунок 2.3 - Рейтинг СКБД на сайті Stack Overflow

Ця система управління базами даних використовує стандартну форму SQL. Утиліти проектування таблиць мають інтуїтивно зрозумілий інтерфейс. MySQL підтримує до 50 мільйонів рядків у таблиці. Граничний розмір файлу

для таблиці за замовчуванням 4 ГБ, але можна збільшити. Підтримує секціонування та реплікацію, а також Xpath та збережені процедури, тригери та уявлення.

- Розробник: Oracle Corporation
- Написана C, C++

Особливості MySQL:

- Масштабованість
- Легкість використання
- Безпека
- Підтримка Novell Cluster
- Швидкість
- Підтримка багатьох операційних систем.

PostgreSQL масштабована об'єктно-реляційна база даних, що працює на Linux, Windows, OSX та деяких інших системах. У PostgreSQL 10 є такі функції, як логічна реплікація, декларативне розбиття таблиць, покращені паралельні запити, безпечніша автентифікація по паролю на основі SCRAM-SHA-256.

- Розробник: PostgreSQL Global Development Group
- Написана на C
- Використовується в компаніях: Apple, Cisco, Fujitsu, Skype та IMDb

Особливості PostgreSQL:

- Підтримка табличних просторів, а також процедур, об'єднань, уявлень і тригерів, що зберігаються
- Відновлення на час (PITR)
- Асинхронна реплікація.

SQLite - це вбудована система, яка реалізує автономний, безсерверний, нульовий конфігурації, механізм транзакції СУБД SQL. Це база даних, яка

налаштована на нуль, що означає, як інші бази даних, які вам не потрібно налаштовувати у вашій системі.

Особливості SQLite:

- Не вимагає окремого процесу сервера або системи для роботи
- Постачається з нульовою конфігурацією, що означає відсутність необхідності в налаштуванні або адмініструванні
- Повна база даних SQLite зберігається в одному крос-платформному диску.
- Дуже маленький розмір, менше 400KB повністю налаштований з додатковими функціями
- Є автономною, що означає відсутність зовнішніх залежностей.
- Транзакції повністю сумісні з ACID, забезпечуючи безпечний доступ до кількох процесів або потоків
- Підтримує більшість функцій мови запитів, знайдених у стандарті SQL92 (SQL2).
- Написана на ANSI-C і надає простий та простий у використанні API
- доступна у UNIX (Linux, Mac OS-X, Android, iOS) та Windows (Win32, WinCE, WinRT).

MongoDB є найпопулярнішою NoSQL системою керування базами даних. Найкраще підходить для динамічних запитів та визначення індексів. Гнучка структура, яку можна модифікувати та розширювати. Підтримує Linux, OSX та Windows, але розмір БД обмежений 2,5 ГБ у 32-бітових системах. Використовує платформи зберігання MMAPv1 та WiredTiger.

- Розробник: MongoDB Inc.
- Написана на C++

Особливості MongoDB:

- Висока продуктивність
- Автоматична фрагментація

- Робота на кількох серверах
- Підтримка реплікації Master-Slave
- Дані зберігаються у вигляді документів JSON
- Можливість індексувати всі поля у документі
- Підтримка пошуку за регулярними виразами.

Microsoft SQL Server є найпопулярнішою комерційною СУБД. Вона прив'язана до Windows, але це плюс, якщо ви використовуєте продукти Microsoft. Залежить від платформи. І графічний інтерфейс, і програмне забезпечення базуються на командах. Підтримує SQL, непроцедурні, нечутливі до регістру та загальні мови баз даних.

- Розробник: Microsoft Corporation
- Написана C, C++

Особливості Microsoft SQL Server**:**

- Висока продуктивність
- Залежність від платформи
- Можливість встановлення різних версій на одному комп'ютері
- Генерація скриптів для переміщення даних.

Порівнявши описані системи обираємо SQLite, бо по-перше на відміну від MongoDB є SQL системою, а цю мову ми вибрали в попередньому пункті «Вибір мов програмування», по-друге вона має розглянуті особливості, які допоможуть нам у створенні нашої аналітичної веб-системи побутового споживання енергії.

2.3 Вибір фреймворку

Фреймворки - це програмні продукти, які спрощують створення та підтримку технічно складних чи навантажених проєктів. Фреймворк, зазвичай, містить лише базові програмні модулі, проте специфічні для проєкту компоненти реалізуються розробником на їхній основі. Тим самим досягається

не тільки висока швидкість розробки, а й велика продуктивність та надійність рішень.

Вебфреймворк – це платформа для створення сайтів та вебдодатків, що полегшує розробку та об'єднання різних компонентів великого програмного проєкту. За рахунок широких можливостей у реалізації бізнес-логіки та високої продуктивності ця платформа особливо добре підходить для створення складних сайтів, бізнес-додатків та вебсервісів.

Для кожної серверної мови програмування є свої фреймворки, так як ми обрали Python тому розглянемо п'ять найпопулярніших фреймворків саме для мови Python.

Рисунок 2.4 - Найпопулярніші фреймворки для Python

Django — безкоштовний веб-фреймворк із відкритим вихідним кодом. Він використовує архітектурний патерн Модель-Представлення-Шаблон (MVT, Model-View-Template).

Особливості Django:

- Маршрутизація URL-адреси
- Міграція схем бази даних
- Робота з базами даних: PostgreSQL, MySQL, SQLite та Oracle.
- Підтримка веб-серверів
- Використання об'єктно-реляційного роутера (ORM)
- Свій двигун шаблонів.

Django має чітку структуру. Фреймворк простий у використанні, тому у програмістів-початківців є всі шанси зрозуміти його і оволодіти ним. Багато особливостей фреймворку дозволяють йому значно спростити та прискорити роботу з кодом. Регулярно оновлюючись він враховує останні версії мови Python.

CherryPy – це мікрофреймворк. CherryPy – це об'єктно-орієнтований HTTP-мікрофреймворк для Python. З його допомогою можна швидко розробляти веб-додатки. Він є компонентом найкращого фреймворку TurboGears для Python.

Особливості CherryPy:

- Дозволяє запускати кілька серверів HTTP одночасно.
- Наявність гнучкої системи плагінів.
- Готові інструменти для аутентифікації, кешування, кодування, сесій, статичного контенту.
- Працює на Android.
- HTTP/1.1 сумісний веб-сервер із пулом потоків WSGI.

Простий та зрозумілий фреймворк, який легко використовувати у роботі. Наявність спеціального плагіна, завдяки якому можна стежити за продуктивністю програми, є плюсом. Фреймворк є безкоштовним та з відкритим вихідним кодом. Програми, розроблені за допомогою CherryPy, працюють на будь-якій операційній системі, яка підтримує Python (Windows, macOS, Linux).

Flask – це ще один мікрофреймворк. Flask має модульну конструкцію. Його також можна використовувати для розробки програм. Він зручний і дозволяє вибирати розширення.

Особливості Flask:

- Наявність вбудованого налагоджувача, власного сервера та безлічі шаблонів.

- Може підключатися до будь-якої ORM.
- Flask має свій двигун Jinja2, але цей фреймворк може працювати і з іншими.
- Заснований на Unicode.
- Сумісний із WSGI 1.0.
- Сумісний із Google App Engine.

Налаштування та встановлення Flask займає менше часу, ніж інших фреймворків. Підтримка безпечних файлів cookie. Вбудовані функції прискорюють розробку. Фреймворк швидко адаптується завдяки модульній конструкції.

Якщо вам потрібно створити щось більш мінімалістичне, вам може сподобатися **Pyramid**. Цей фреймворк підходить для будь-яких проектів. У ньому є корисні функції для створення складних програм або масштабування невеликих сайтів під підвищене навантаження.

Особливості Pyramid:

- Зручні інструменти для роботи із статичними активами.
- Предикати та рендеринг.
- генерація URL.
- Підтримка SQLAlchemy.
- SQLAlchemy забезпечує комфортну роботу з базою даних навіть за складних запитів.

Pyramid відрізняється гнучкістю та простотою налаштування. Будь-який компонент фреймворку, будь то база даних або шаблонизатор, може бути замінений. Можна навіть використовувати кілька різних компонентів одночасно (наприклад, підключити дві різні бази даних).

Також доступні зручні Ajax-запити. Завдяки системі декораторів та уявлень можна відправляти XHR-запити без будь-яких додаткових зусиль з боку розробника.

Web2py – спеціальний стековий фреймворк. Насамперед він хороший тим, що його не потрібно встановлювати. Його можна легко запустити з будь-якого пристрою – флешки чи жорсткого диска.

Особливості Web2py:

- Власна IDE з редактором коду, відладчиком і розгортанням.
- Трекер помилок.
- Функція зворотної сумісності дозволяє працювати з попередніми версіями фреймворку.
- Якісний захист даних.
- Відкритий вихідний код.
- Гнучкість та безкоштовність.

Може працювати на різних платформах і з різними протоколами, не потребує встановлення та налаштування.

Отже, існує багато корисних популярних фреймворків Python для оптимізації процесу та прискорення часу розробки. Існують повностекові та неповностекові фреймворки, а також мікрофреймворки для розробки на Python.

Основна перевага повностекових фреймворків у тому, що в них вже є все необхідне для повноцінного застосування. Немає необхідності шукати окремі бібліотеки для кожного дрібного завдання і думати про доступ до бібліотек, обробку запитів та сумісності, тому ми обираємо фреймворк Django де навіть новачки можуть швидко створити готову робочу вебсистему.

В підсумку маємо обраний стек технологій для розробки нашої веб-системи, а саме:

- HTML, CSS, JavaScript – для створення фронтенду
- Python – для розробки бекенду
- SQLite – для створення бази даних
- Django – безкоштовний веб-фреймворк.

РОЗДІЛ 3. РОЗРОБКА МОДЕЛЕЙ ТА WEB-ПОРТАЛУ

3.1 Розробка ER моделі

Модель «сутність-зв'язок» (ER-модель) (англ. Entity-relationship model або entity-relationship diagram) — модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків. ER-модель — це мета-модель даних, тобто засіб опису моделей даних. Існує ряд моделей для представлення знань, але одним з найзручніших інструментів уніфікованого представлення даних, незалежного від програмного забезпечення, що його реалізує, є модель «сутність-зв'язок». Важливим є той факт, що з моделі «сутність-зв'язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об'єктна), тому вона є найзагальнішою [30].

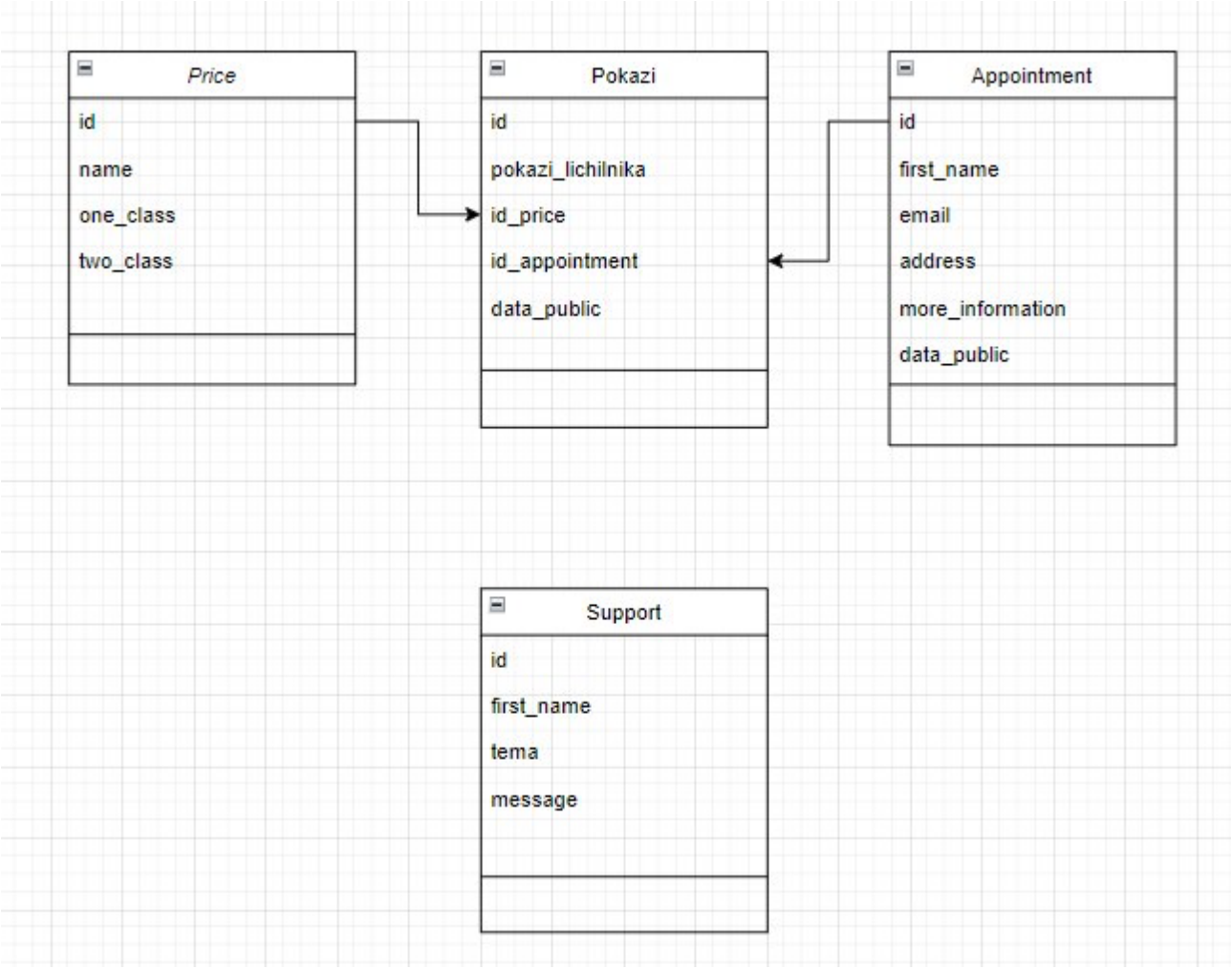


Рис. 3.1 - Модель «сутність-зв'язок» (ER-модель)

3.2 Розробка UML діаграми класів

Для розробки та візуалізації артефактів програмної системи стандартною мовою є UML. Взаємозв'язок між різними об'єктами описується діаграмою класів, яка забезпечує дизайн та аналіз програми та розглядає її у статичній формі. Будучи найважливішою діаграмою UML, діаграма класів складається з класу, атрибутів та зв’язків, які є її істотними елементами. Щоб отримати уявлення про структуру програми, використовується діаграма класів, яка допомагає скоротити час ознайомлення [31].

- верхня секція містить назву класу;
- середня секція описує стан, тому містить перелік полів класу, імена та типи даних, разом із указанням області видимості (“-“ – private, “+” – public, “#” – protected);
- нижня секція описує поведінку, тому містить містить перелік методів – імена методів, їх тип (статичні чи нестатичні), тип повернення.

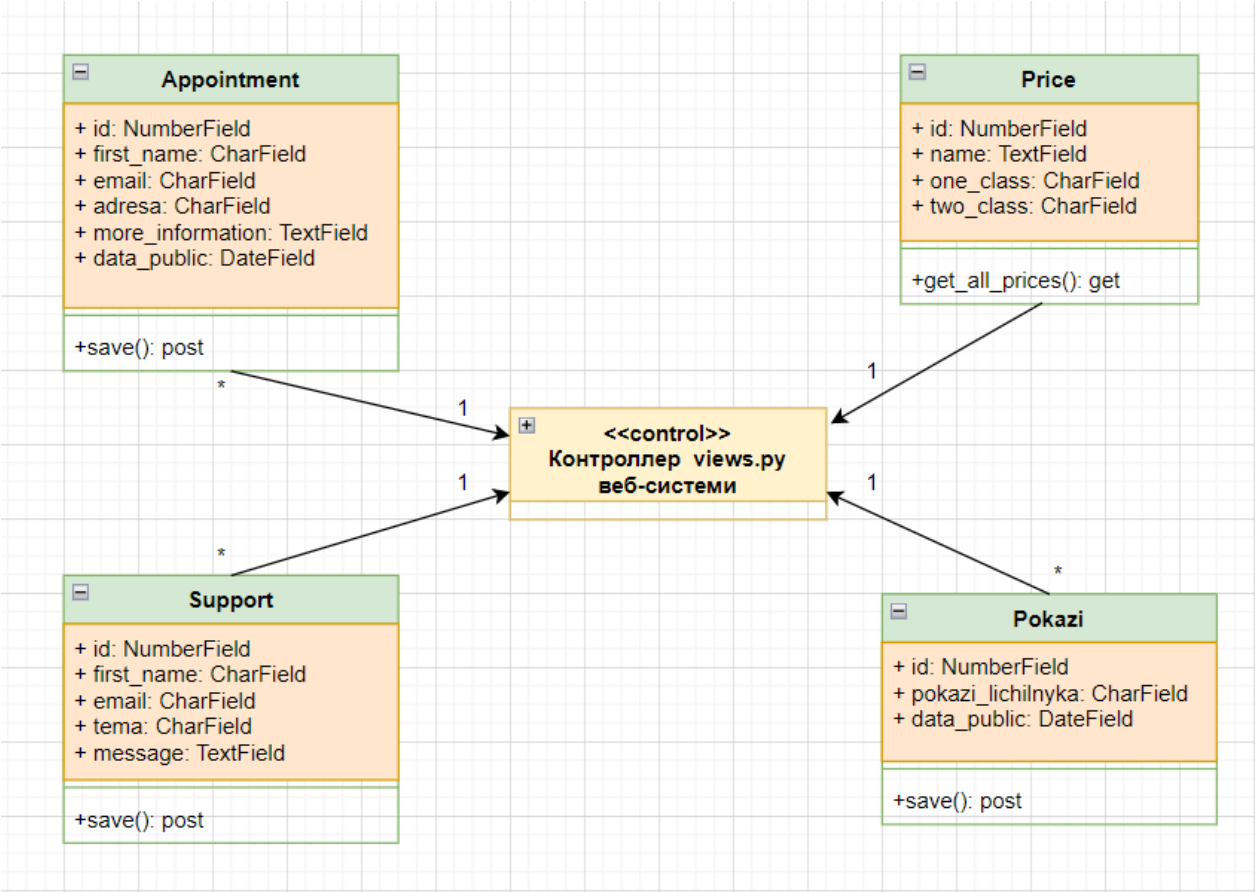


Рис. 3.2 - UML діаграма класів

3.3 Дизайн вебсайту системи побутового споживання енергії

Розробка дизайну сайту починається з прототипування. Це схематичне розташування блоків. Прототипування робиться для того, щоб можна було побачити головні елементи сторінок, структуру сайту [32].

Багато помилок, які виникають на початковому етапі розробки того чи іншого проєкту, пов'язані з неповним розумінням проєкту. Упускаються багато деталей з уваги, які згодом можуть дорогого коштувати. Для того щоб уникнути або мінімізувати ці помилки, і проводиться прототипування. Прототипування сайтів дає можливість швидко побудувати чітку структуру, проаналізувати її. Виявити всі недоліки і в найкоротші терміни усунути, з мінімальними втратами часу та фінансових коштів.

Дехто вважає, що прототипування не так важливо, оскільки багато проєктів схожі і схема розташування блоків та елементів однаково. Насправді це частково не так, оскільки споживачі відрізняються один від одного, і мислять вони по-різному. Потрібно враховувати цільову аудиторію, як і що вона сприймає, на чому звернути увагу, а на чому ні.

На етапі прототипування сайту деталізація інтерфейсу та інших дрібниць не дуже важливі, важливо побудувати правильну структуру та зрозуміти, як вона взаємодіятиме з користувачами.

Також не варто забувати про те, що дизайнеру буде набагато простіше розробити дизайн сайту по прототипу.

До прототипів зазвичай застосовують такі вимоги як:

- Висока швидкість створення прототипу. Швидке прототипування дозволяє заощадити час на розробку проекту.
- Деталізація (не всім видів прототипів).
- Легкість та швидкість внесення змін до прототипу.
- Інтерактивність (можливість прототипу реагувати на дію користувача).

- Доступність для всіх учасників процесу (замовник, дизайнер, менеджер, програміст тощо).

Методи прототипування сайтів

Існують декілька різних методів прототипування сайтів, з яких виділяємо основні:

- Паперове прототипування;
- Прототипування за допомогою спеціальних програм (Axure Pro, WireframeSketcher, SketchFlow тощо);
- Прототипування за допомогою графічних програм (Photoshop, Illustrator, InDesign тощо).

Паперове прототипування

Один з найпростіших і найшвидших методів створення прототипів сайтів, зі своїми плюсами та мінусами. Для того щоб створити прототип достатньо взяти аркуш паперу та ручку (можна кілька кольорів для наочності) або олівець.

З плюсів даного методу можна відзначити:

- високу швидкість створення прототипу;
- можливість створювати досить детальні прототипи;
- не потрібні спеціальні знання (програмування, володіння графічними редакторами);
- оптимальна швидкість зміни прототипу за рахунок перемальовки;
- можливість залишати коментарі у будь-яких місцях прототипу будь-яким із учасників розробки проекту.

З мінусів даного методу можна відзначити:

- відсутня інтерактивність у прототипу, може виявити ряд проблем в інтерфейсі;
- внесення змін не завжди можливе, доводиться перемальовувати;
- недостовірність розмірів елементів на прототипі, можливі наслідки створення дизайну сайту;

- не завжди має естетичний вигляд, за рахунок чого може не сподобатися замовнику.

Прототипування за допомогою спеціальних програм (Axure Pro)

Наступний метод ми розглянемо однією з найпопулярніших програм для прототипування сайтів Axure Pro. Досить проста програма, в якій сторінки організовані у деревоподібній формі. Дуже гнучка та зручна в обіг програма. Швидкість роботи значно збільшена завдяки можливості повторного використання об'єктів (форми, картинки, текст тощо.). Кожен об'єкт має властивості, які можна змінювати. Прототипи виходять естетичними та інтерактивними. Швидкість зміни досить висока, за рахунок можливості створення своїх об'єктів конкретних областей, будь то меню, хедер, футер, які легко можна повторно використовувати. Можливість часткового юзабіліті тестування.

З плюсів даного методу можна відзначити:

- високу швидкість створення прототипу;
- висока деталізація прототипу;
- прототипи мають естетичний вигляд;
- прототипи із середньою інтерактивністю;
- висока швидкість внесення змін без повторного відтворення прототипу;
- можливість створення прототипу у вигляді Html чи картинки;
- повна доступність всім учасників розробки проекту.

З мінусів даного методу можна відзначити:

- мінусів не виявлено, хіба вивчення програми.

Якщо прототип влаштовує, тоді розробляється повноцінний дизайн. І вже після прототипування вебдизайнер змальовує макети.

Прототипування за допомогою графічних програм (Photoshop [23])

Наступний спосіб пропоную розглянути на прикладі програми Adobe Photoshop [12]. Вибір мій зупинився на цій програмі через свій особистий

досвід прототипування сайтів. Особисто я створюю прототипи в основному на папері та у фотошопі. Так як у фотошопі працюю постійно, програму знаю досить непогано, тому створення прототипу в ній у мене не викликає проблем. Прототип можна створити досить швидко і зрозуміло, естетично привабливо. Більше такий метод підійде для створення простих та нескладних прототипів. Для об'ємних і складних проектів все ж таки більш прийнятно використовувати спеціалізовані програми.

З плюсів даного методу можна відзначити:

- середня швидкість створення прототипу;
- висока деталізація прототипу;
- прототипи мають естетичний вигляд;
- висока швидкість внесення змін без повторного відтворення прототипу;
- повна доступність всім учасників розробки проекту.

З мінусів даного методу можна відзначити:

- відсутня інтерактивність у прототипу, створеного у фотошопі, може виявити ряд проблем в інтерфейсі. У деяких програмах, таких як InDesign, можна створити низьку інтерактивність;
- потрібні знання з користування графічним редактором.

Отже, прототипування нашого вебсайту ми будемо робити за допомогою Adobe Photoshop.

Детально розглянемо всі етапи макетування нашої головної сторінки. Розіб’ємо макетування index.html на три етапи - блок «Header», блок «Наша спеціалізація», та об’єднаний блок «Наші сервіси» з «підвалом» сайту.

Спочатку визначаємо всі основні елементи блоку «Header» — почнемо зі створення білого прямокутника, щоб намітити межі. Тут використовується інструмент Rectangle (1900 x 900 пікселів).

Рис. 3.3 - Використання інструменту Rectangle

Далі темним прямокутником визначимо панель навігації. Слід врахувати, що згодом на місці великого білого прямокутника буде блок зі змінним зображенням.

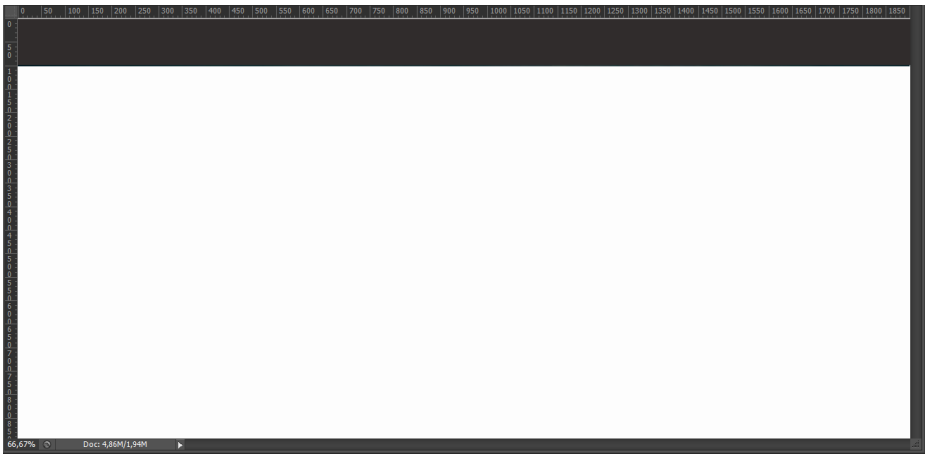


Рис. 3.4 - Прототипування панелі навігації

Наступним кроком додаємо заголовок та елементи навігації в макет сторінки.

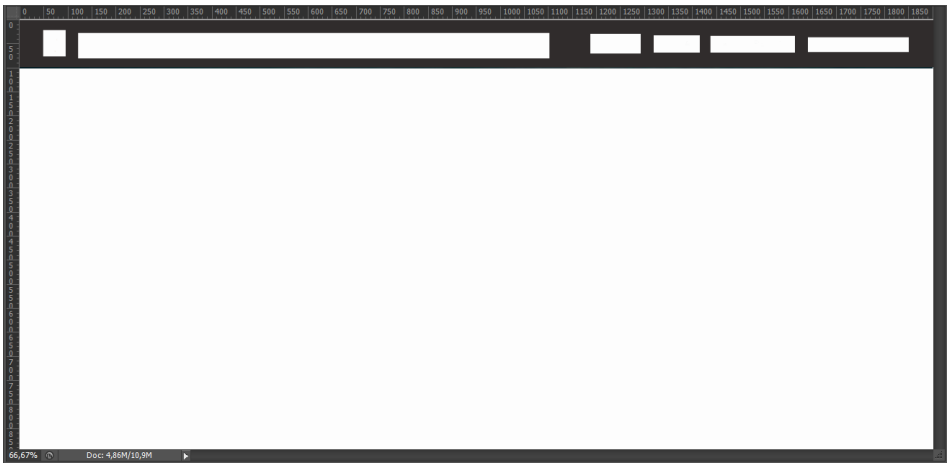


Рис. 3.5 - Прототипування заголовку та елементів навігації

Далі темними прямокутниками додаємо всі місця для написання тексту в нашій «шапці» сайту.

Рис. 3.6 - Додавання місць для написання тексту

Додаючи місце для трьох кругів та двох перемикачів зображень, використаємо інструмент Rounded Rectangle. Даний інструмент дозволяє змінити радіус кута фігури будь-якої миті. Це означає, що ви можете повернутися і змінити кути пізніше, якщо ваш дизайн вимагатиме цього. Також додаємо місце для кнопки з текстом.

Рис. 3.7 - Використання інструменту Rounded Rectangle

Далі за аналогією створюємо блок «Наша спеціалізація», враховуючи, що навігаційна панель рухається при скролі миші, тому також відображаємо її при протопипуванні блоку.

Рис. 3.8 - Створення прототипу блоку «Наша спеціалізація»

І наостанок прототипуємо об’єднаний блок «Наші сервіси» з «підвалом» сайту.

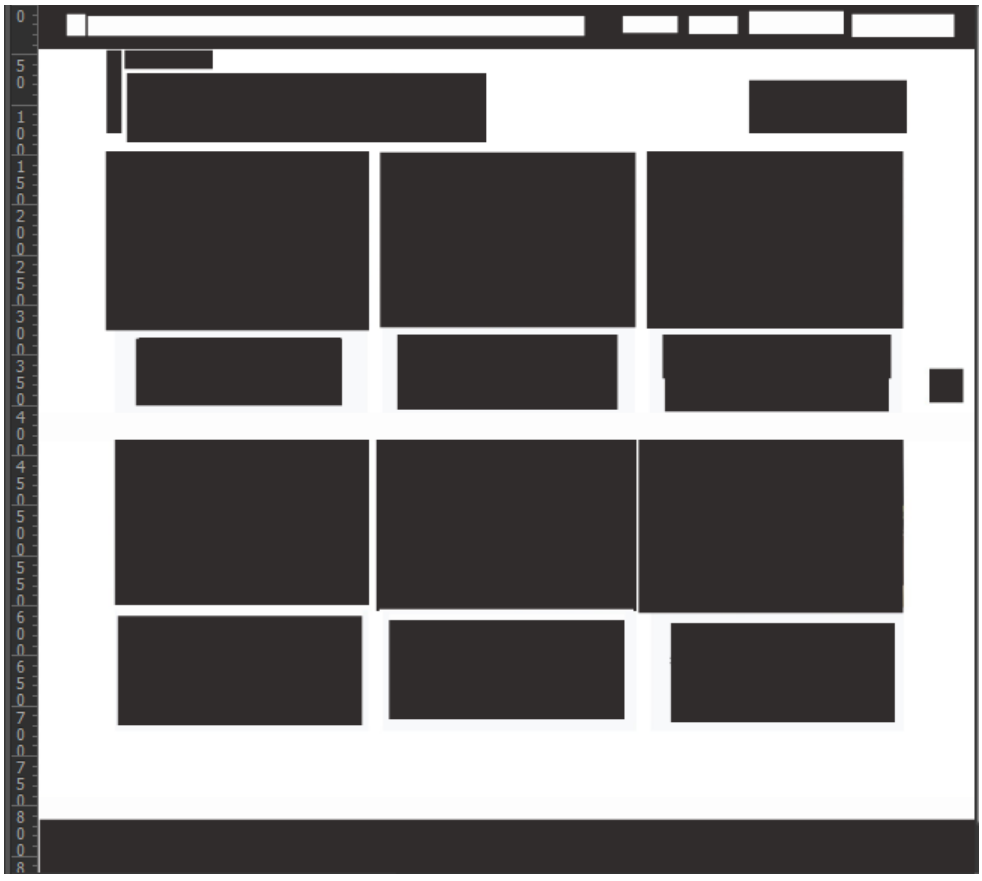


Рис. 3.9 - Створення прототипу блоку «Наші сервіси»

Прототипування всі інших сторінок вебсайту робимо за аналогією. Після того, як усі макети готові переходимо безпосередньо до дизайну по створеним прототипам вебсайта.

Етапи створення дизайну по макету виглядають аналогічно прототипуванню, знову ж розглянемо на прикладі нашого сайту.

При створенні дизайну вебсайту необхідно враховувати психологію кольору в дизайні основні правила його використання [17].

Колір не менш важливий, ніж якість верстки та ілюстрацій, вибраний шрифт або зміст написаного повідомлення. Більше того, він виступає частиною цього повідомлення, і тому важливо правильно підбирати колір, виходячи з ідей, цілей та послання бренду. Далі ми розглянемо психологічне значення кольору і наведемо приклади того, як дизайнери та маркетологи їх використовують.

Правила кольору у дизайні [18]

Вибір кольору залежить від продукту чи послуги

Люди звертають увагу на колір перед придбанням продукту чи послуги, і дизайнер має це використовувати. Наприклад, наголошувати на головній перевагі товару відповідним кольором.

Вибір кольору може залежати від статі аудиторії

Дизайнер Джо Хеллок провів дослідження і уклав — те, наскільки колір може залежати від статі аудиторії. З'ясувалося, що чоловіки і жінки люблять синій, і мало хто любить помаранчевий. У той же час фіолетовий колір викликає більшу симпатію у жінок, ніж у чоловіків.

Треба враховувати, що це лише один із факторів, який можна використовувати у виборі кольору бренду чи кампанії. До того ж, далеко не визначальний — значно важливіша залежність кольору від повідомлення, яке ви хочете донести.

Також Джо Хеллок досліджував, наприклад, залежність симпатій до певних кольорів у випробуваних різного віку. Коротко: з роками люди все більше люблять синій, а ось коричневий із помаранчевим ні в кого не в пошані.

Ще слід враховувати вплив на психологічний стан людини теплих та холодних кольорів.

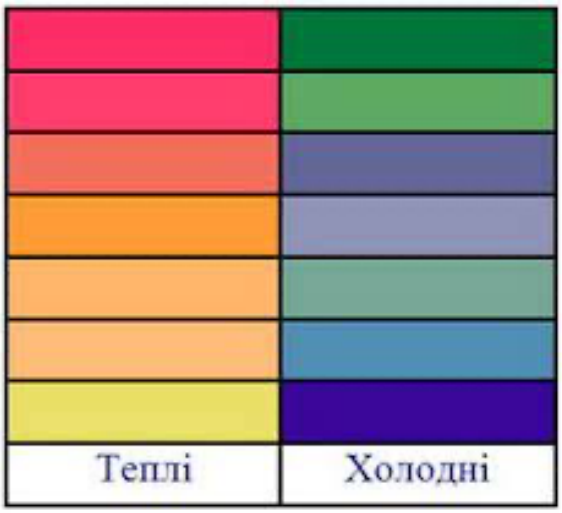


Рис. 3.10 - Теплі та холодні кольори

Колір робить продукт пізнаваним — іноді більше, ніж інші елементи дизайну. Наш мозок фокусується на відомих брендах, тому дизайнер повинен приділяти особливу увагу домінуючому кольору, який довго використовуватиметься в айдентиці компанії.

Це твердження легко довести. Коли ми перебуваємо в незнайомому районі чи чужій країні і шукаємо, де поїсти чи випити каву, ми можемо побачити «Макдональдс», KFC або, наприклад, Starbucks здалеку — задовго до того, як розглянемо назву чи логотип закладу. Все тому, що брендам вдалося "закріпити за собою" кольори, які міцно з ними асоціюються.

Нижче наведені основні ассоціації які викликає у людини певний колір.

Червоний

Позитивні асоціації: сила, збудження, пристрасть, енергія, молодість.

Можливі негативні асоціації: гнів, небезпека.

Помаранчевий

Позитивні асоціації: довіра, тепло, інновації, доброзичливість, енергія, хоробрість.

Можливі негативні асоціації: розчарування, невігластво, незрілість.

Жовтий

Позитивні асоціації: оптимізм, тепло, щастя, креативність, доброзичливість.

Можливі негативні асоціації: обережність, неспокій, страх.

Зелений

Позитивні асоціації: здоров'я, надія, природа, зростання, свіжість, добробут.

Можливі негативні асоціації: заздрість, хвороба, нудьга.

Синій

Позитивні асоціації: віра, відданість, надійність, логіка, спокій, безпека.

Можливі негативні асоціації: холод, беземоційність, байдужість.

Фіолетовий

Позитивні асоціації: мудрість, розкіш, добробут, духовність, витонченість, велич.

Можливі негативні асоціації: зосередженість собі, занепадництво, пригніченість.

Рожевий

Позитивні асоціації: уява, запал, перетворення, баланс, креативність.

Можливі негативні асоціації: надмірна жіночність, імпульсивність.

Чорний

Позитивні асоціації: витонченість, безпека, сила, влада, твердість.

Можливі негативні асоціації: пригніченість, холод, загроза, трагічність.

Білий

Позитивні асоціації: чистота, ясність, непорочність, простота, свіжість.

Можливі негативні асоціації: стерильність, холод, ізоляція.

Враховуючи вплив кольорів на людину, спочатку створимо навігаційну панель вибравши колір тла білим, активний текст та зображення помаранчеві, а неактивний текст чорний.

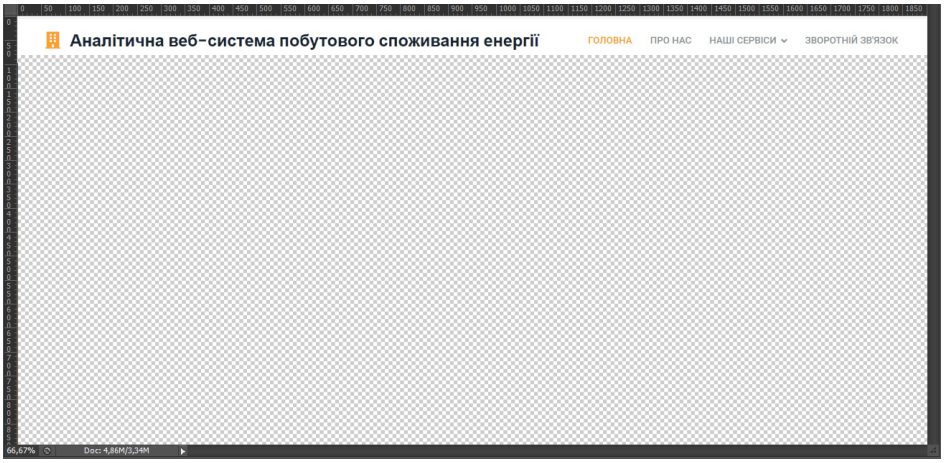


Рис. 3.11 - Створення дизайну навігаційної панелі

Далі додамо до нашого дизайну зображення, незабуваємо, що воно буде змінним.

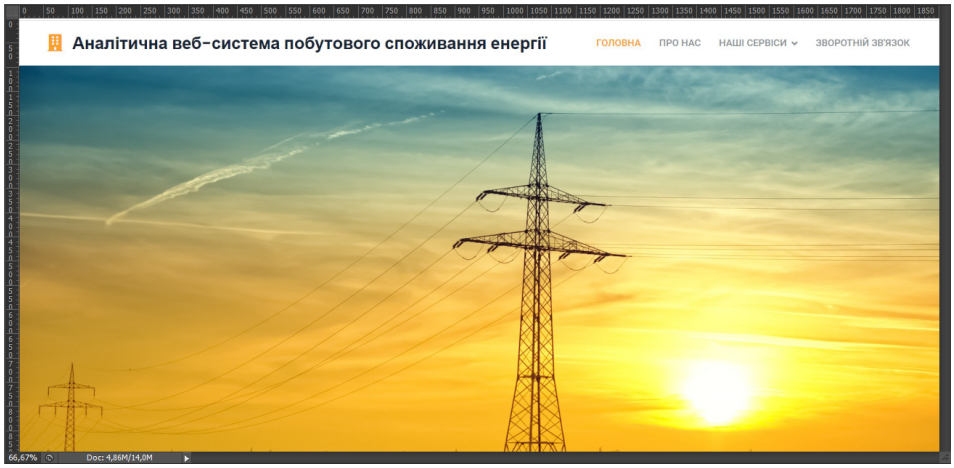


Рис. 3.12 - Додавання до дизайну змінного зображення

І наостанок додаємо текст та інші елементи.

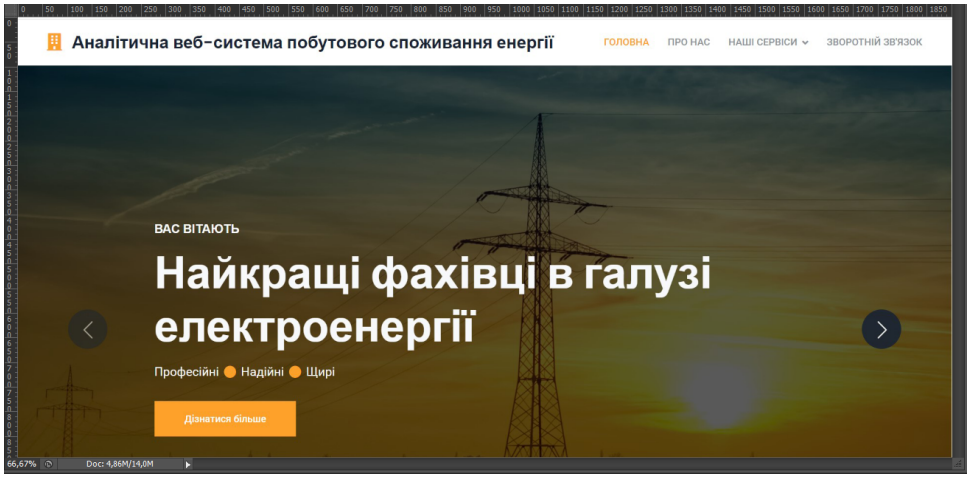


Рис. 3.13 - Додавання до дизайну тексту

Далі, аналогічно, робимо дизайн блоку «Наша спеціалізація».

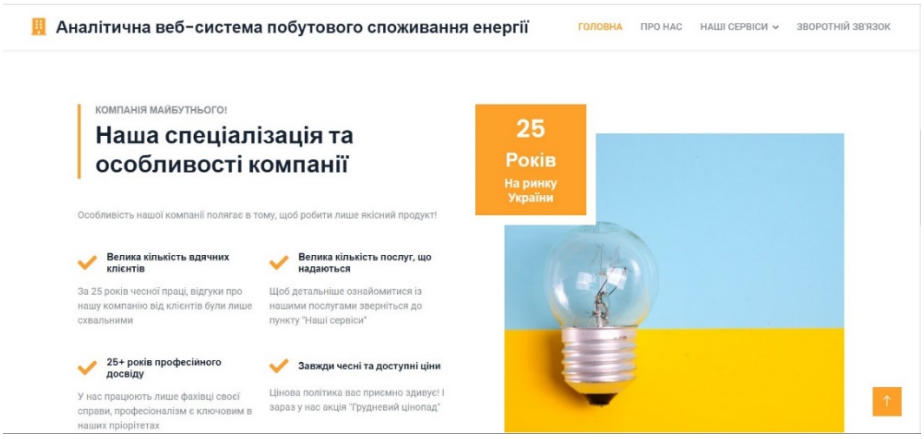


Рис. 3.14 - Дизайн блоку «Наша спеціалізація»

Та блоку «Наші сервіси» разом з футером, «підвалом» вебсайту.

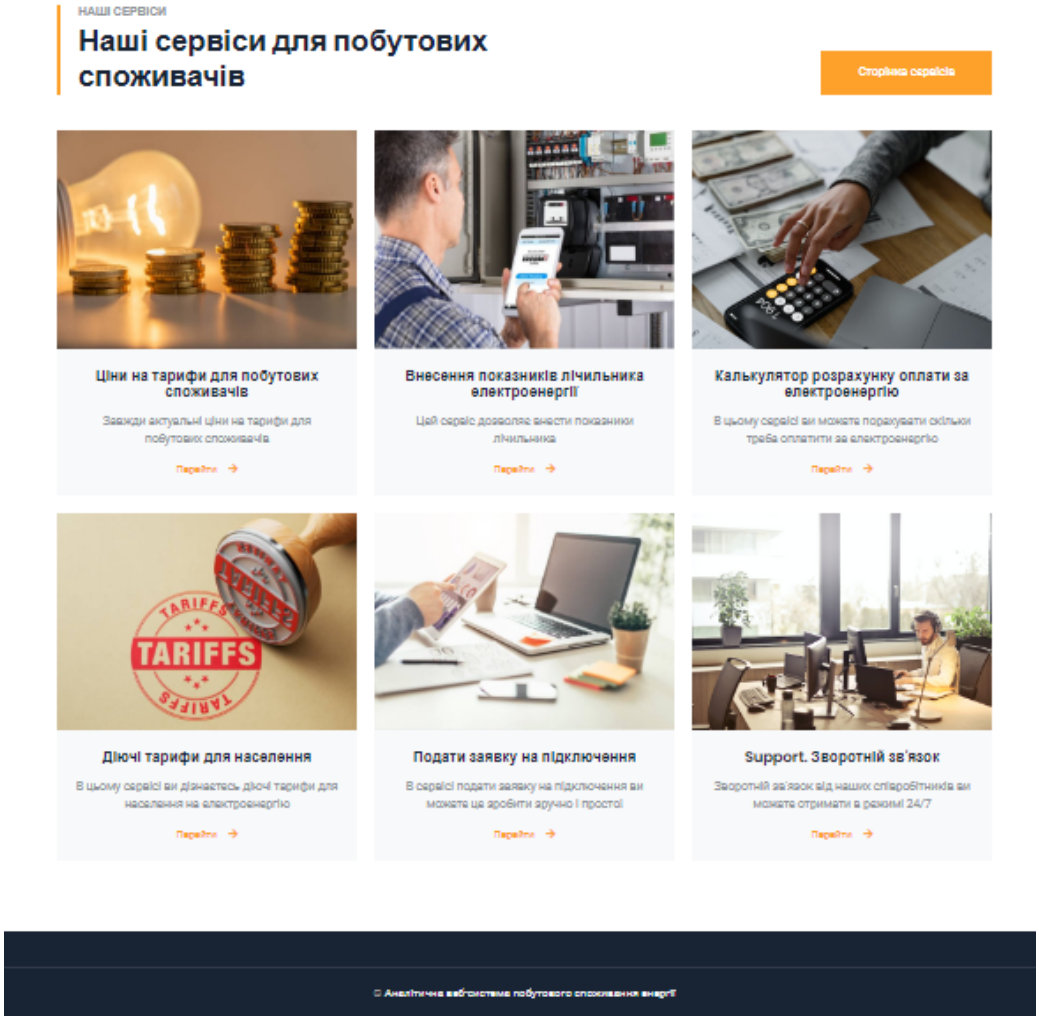


Рис. 3.15 - Дизайн блоку «Наші сервіси»

За аналогією, створюємо дизайн всії сторінок сайту. Створивши дизайн для всіх сторінок, верстка сайту, буде менш складним процесом.

Дизайн всієї головної сторінки буде мати наступний вигляд.

Рис. 3.16 - Дизайн головної сторінки сайту

3.4 Архітектура вебсайту

Архітектура сайту є структурою його сторінок і програмної частини. Це певна, чітко оформлена система, що організовує різні файли, гіперпосилання, заголовки, пошук, навігаційну панель і просто інформацію, яка міститься на сайті. Саме логічна побудова всіх цих елементів, їхнє місце розташування та взаємозв'язок визначають, наскільки буде зручно і просто користуватися цим ресурсом. Така структура дає можливість користувачеві бачити всі розділи сайту, дозволяє ознайомитися з різною інформацією, представленою на ньому, тобто значно підвищує юзабіліті. Створення архітектури сайту є першочерговим етапом у його проектуванні [32].

Архітектура сайту потрібна для створення ресурсу, який допоміг би абсолютно будь-якому користувачеві знаходити потрібну йому інформацію, не витрачаючи на це великої кількості часу. Навіть за умови, що він опинився на сайті вперше, користувач повинен швидко освоїтися і зрозуміти, як розташована інформація на даному майданчику. Сучасна людина малоймовірно вирішить витрачати надто багато часу, відкриваючи нескінченну кількість посилань у пошуках потрібної інформації. У цій ситуації користувачеві зручніше залишити сайт і просто ввести запит у будь-яку пошукову систему.

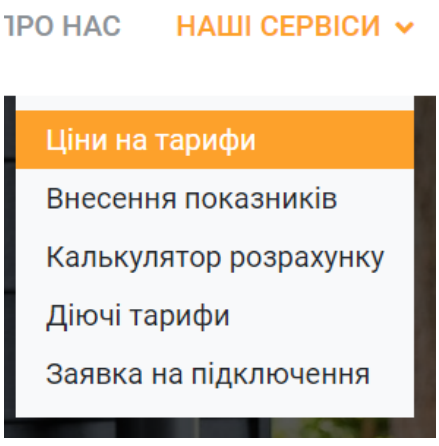


Рис. 3.17 - Архітектура блоку «Наші сервіси»

Добре продумана та прописана архітектура як великого, так і маленького веб-сайтів здатна значно спростити індексацію сторінок. Проте реалізовувати

цю концепцію набагато зручніше на початковому етапі проектування ресурсу, коли вносити зміни ще не є чимось складним.

Будь-якій людині знайома ситуація, коли пошук потрібної інформації на створеному сайті є досить складним і довготривалим процесом, особливо, якщо ресурс багаторазово збільшив кількість своїх сторінок. Частковим виходом із ситуації може стати пошук за ключовими словами, проте таке рішення ніяк не впливає на індексацію, та й є оптимальним далеко не для всіх.

```
<meta content="Аналітична, веб-система, споживання,
енергії " name="keywords">
```

Тому оптимізація архітектури сайту є в цьому випадку найзручнішим рішенням. Це забере багато часу, але зробить сайт найбільш зрозумілим простому користувачеві, йому буде легше орієнтуватися і шукати необхідну інформацію.

Процес оптимізації структури являє собою систематизація всіх сторінок, грамотний розподіл внутрішніх посилань, робота з необхідними тегами, які є незамінними при пошуковому просуванні. Основна мета оптимізації в цьому випадку – зробити сайт максимально корисним та привабливим для користувача, допомогти йому знаходити потрібну інформацію, докладаючи мінімум зусиль.

Якщо ж справа стосується комерційного ресурсу, то можуть з'явитися інші вимоги, наприклад, зробити так, щоб для відвідувачів стали найбільш доступними ключові сторінки, які приносять основний прибуток.

Добре розроблена архітектура сайту допомагає виділити найважливіші сторінки для пошукових систем, наприклад, «Сервіси» або «Послуги». Такі посилання зазвичай розміщені на головній. Цей факт привертає увагу різних пошукових систем, отже, забезпечує найвищі позиції результаті пошуку.

3.5 Опис файлів, функцій та процесів

Як ми визначили в попередньому пункті дуже важливим кроком розробки є створення правильної, гнучкої та масштабованої архітектури. На рисунку 3.18 зображена структура проєкту.

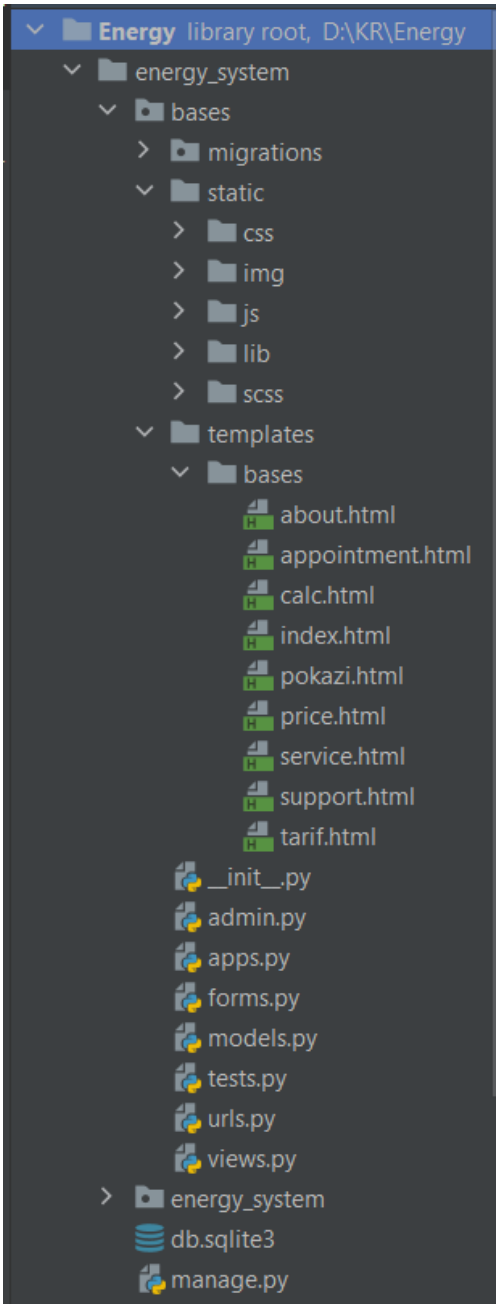


Рисунок 3.18 - Структура проєкту

В таблиці 3.1 наведено опис та призначення головних файлів в структурі проєкту.

Таблиця 3.1 Опис та призначення головних файлів в структурі проєкту

Файл або Папка	**Опис**
Energy_system	Папка з основним проєктом
Bases	Папка з базовим проєктом
Migrations	Папка з міграціями
Static	Папка із статичними файлами і папками. Каскадними таблицями стилів css, зображеннями img, javascript скриптами js, бібліотеками lib, та препроцесами scss.
Templates	Папка із html-файлами
Admin.py	Файл із налаштуваннями адмін-панелі
Forms.py	Файл із налаштуваннями html-форм
Models.py	Файл із налаштуваннями моделей бази даних
Urls.py	Файл із налаштуваннями роутів
Views.py	Файл-контроллер

Як можна побачити з таблиці 3.1, структуру проєкту можна поділити на дві основні частини - вихідні файли, які групуються завдяки реалізації шаблону MTV (Model-Templates-Views), та веб-частини, яка дозволяє нам мати візуальний інтерфейс веб-системи в будь-якому браузері.

Всі ці файли необхідні для розгортання сайту на сервері Django (або веб-хостингу) та її запуску.

3.6 Верстка вебсайту. Frontend

Термін «верстка» прийшов у веб із видавничого ремесла. Там під версткою розуміється монтаж макета газети чи книги з підготовленого матеріалу — тексту, таблиць, зображень, що потім вирушає до друку. В обох випадках головне завдання верстки — зробити так, щоб сайт чи друковане видання було зручно переглядати та взаємодіяти з ними.

Верстка - це створення структури гіпертекстового документа і розробка графічного інтерфейсу користувача за допомогою HTML, скриптів і стилів. Головне завдання верстальника – точно і без помилок відправити прототип до реального коду. Простими словами, верстка - це перенесення попереднього макету сайту в HTML.

Англомовний еквівалент терміна «верстка веб-сторінок» — Frontend web development, що можна перекласти як «фронтенд веб-розробки». Фронтенд сайту це те, що бачить звичайний користувач - текст, зображення, інші графічні елементи. Бекенд сайту – це внутрішня, невидима оку користувача частина сайту – як підкапотний простір автомобіля.

Верстка розташовує всі елементи на сторінці і робить так, щоб з ними було зручно працювати. Тому верстка сайту – це відповідальне завдання, яке вимагає уважності, терпіння та постійного тестування.

Верстку веб-сторінок неможливо уявити без HTML [19]. Якщо говорити простими словами, HTML це єдиний стандарт відображення всіх елементів веб-сторінки. Це мова розмітки, за допомогою якої браузери показують нам порядок, розмір, форми та шрифт тексту. З його тегами знайомі всі, хто займався створенням сайтів, наприклад:

<body> </body> - весь вміст сторінки;

<h1> </h1> - це позначення заголовка;

<h2> </h2> - підзаголовок;

<img> - зображення;

 - жирний шрифт;

Етапи верстки сайту

Залежно від типу завдання та її складності, етапи верстки сайту можуть трохи відрізнятися. Стандартний алгоритм роботи верстальника має такий вигляд:

- Перенесення структури попереднього макету в HTML-документ.
- Робота зі «стилями», завдання глобальних налаштувань CSS згідно з параметрами прототипу.
- Налаштування та завдання властивостей для інтерактивних компонентів сторінки. При цьому можуть використовуватися різноманітні технології – наприклад, JQuery та JS (і, звичайно, «стилі»).
- Перенесення попереднього шаблону до системи керування контентом.
- Тестування. На цьому етапі розробник намагається виявити типові проблеми: наприклад, некоректне відображення сторінок у різних браузерах або з'їзд блоків різних мобільних пристроях.

Сьогодні найпоширенішими є два типи верстки: блокова та таблична.

Блокова верстка

Блокова верстка цілком відповідає основним принципам адаптивності і ним не суперечить. Сучасний сайт краще верстати саме за її допомогою.

Більш сучасна і чуйна, добре поєднується зі стилями вже «з коробки». Це означає, що за необхідності можна швидко змінити розташування об'єктів, задати потрібні відступи, інтервали, змінити рішення кольору будь-якого елемента сторінки. Крім того, блокові елементи сторінки краще з точки зору швидкодії та чуйності.

Схематично уявити блоковий тип верстки можна так:

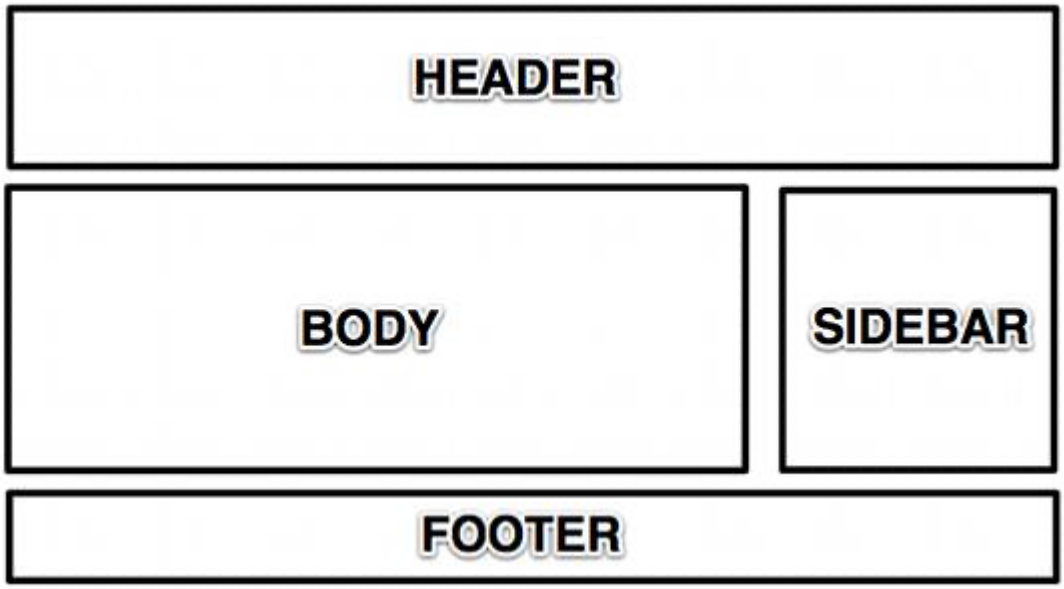


Рисунок 3.19 - Блоковий тип верстки

Таблична верстка

Застаріле рішення, яке активно використовувалося в інтернеті десятки років тому. Сьогодні цей підхід розмітки під час створення сайтів практично не зустрічається.

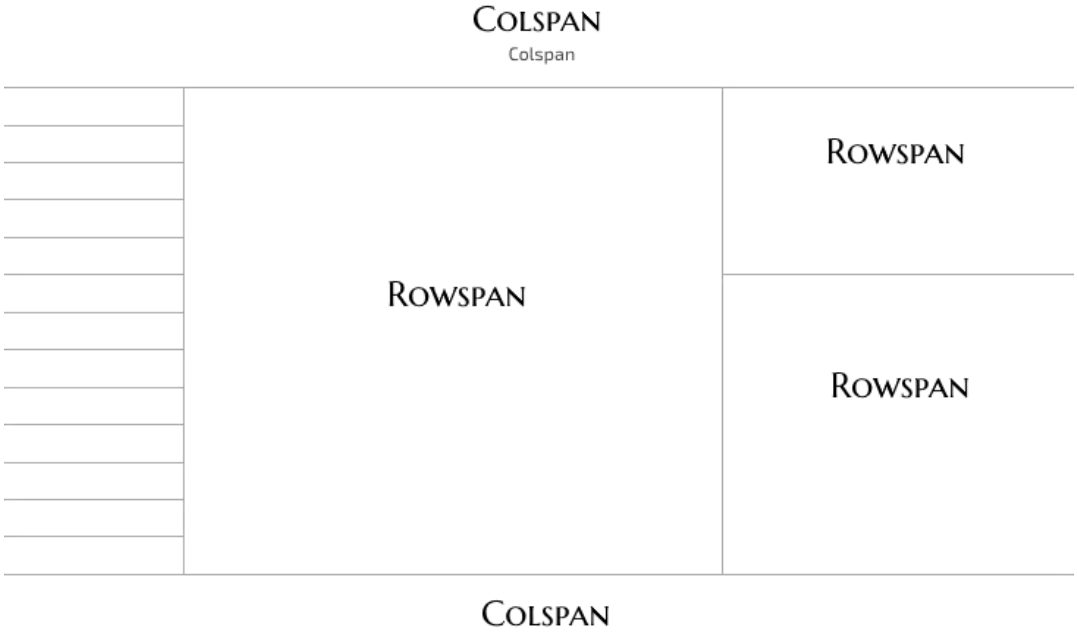


Рис. 3.20 - Табличний тип верстки

На відміну від блочного підходу, в табличному, розмітка задається тегом <table>. Після нього в ієрархічному порядку слідує <tr> і <td> — це рядки та стовпці, відповідно.

Розмістити елемент на сторінці можна тільки створюючи порожні комірки, які будуть подібними до скелета. Такий підхід є досить трудомістким. Крім того, він збільшує фізичний розмір сторінки, що завжди погано.

Таблична верстка не підходить для сучасних сайтів, що проектуються на основі адаптивного дизайну. Мабуть, єдине вдале застосування табличної верстки - це гумовий дизайн сторінок (коли сторінка відкривається на весь екран незалежно від дозволу).

Для верстки вебсайту необхідно володіти наступними технологіями це мова гіпертекстової розмітки HTML [19], каскадні таблиці стилів CSS [21], та мову програмування Javascript [10].

Треба чітко розуміти що HTML – це каркас сайту [24]. По ньому браузер малює веб-сторінку, наводить красу за допомогою CSS і додає інтерактив через JavaScript.

Отже, для того щоб зручно використовувати наведені технології потрібен якийсь програмний продукт, який буде легко співпрацювати у їх поєднанні.

Тому, для створення верстки по дизайну обираємо Bootstrap [27], найпопулярніший у світі HTML, CSS и JS фреймворк для створення адаптивних сайтів, а також програму-редактор Notepad++.

Bootstrap з'явився далекого 2011 року в компанії Twitter. Спочатку це була невелика бібліотека для внутрішніх потреб компанії, проте він швидко завоював популярність і вийшов далеко за її межі.

На сьогоднішній день Bootstrap є визнаним фаворитом серед розробників адаптивних сайтів. Це фреймворк, що складається із шаблонів HTML, CSS та розширень JavaScript із бібліотекою JQuery [13-14].

Він дотримується принципу "mobile-first", що передбачає початкову розробку сайту для невеликого екрана мобільного телефону, потім контент розміщується згідно з роздільною здатністю планшета і, нарешті, верстальник працює над версією для ноутбука та комп'ютера. Завдяки такому підходу

кінцевий користувач може переглядати сайт на будь-яких пристроях, не відчуваючи найменшої незручності.

Існує два способи підключення фреймворку до Вашого проекту. Його можна завантажити з офіційного сайту http://getbootstrap.com або використовувати посилання на версію CDN.

Використання CDN має свої плюси та мінуси. Безперечною перевагою є швидке завантаження. Однак, якщо мережа недоступна, фреймворк не завантажиться.

- Переваги сайту на Bootstrap
- Швидкість розробки
- Адаптивність
- Популярність
- Безкоштовність
- Гнучкі налаштування
- Документованість
- Підтримка

Bootstrap активно розвивається, виходять нові версії, виправляються помилки та з'являється новий функціонал.

Коротко розглянемо принцип роботи фреймворка на прикладі компонента Dropdowns [27]:

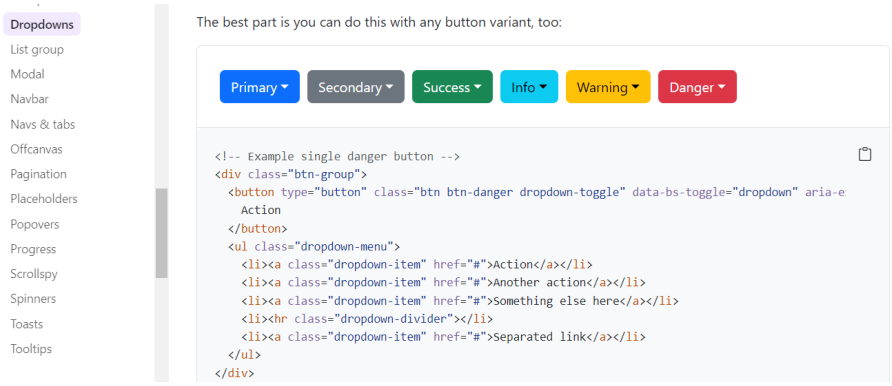


Рис. 3.21 - Компонента Dropdowns Bootstrap

Вся система стилів побудована на класах, тобто якщо ми хочемо, щоб елемент сприймався бібліотекою як кнопка, тоді потрібно прописати клас btn.

Якщо хочемо, щоб кнопка була червоного кольору, тоді додаєм ще один клас btn-danger, якщо жовтого btn-warning.

Сайти створені за допомогою Bootstrap мають однакову навігацію, структуру, кнопки. Щоб вирішити проблему, можна змінювати шаблон в залежності від ідей дизайнерів та побажань замовника.

Приведемо приклад навігаційної панелі файлу index.html:

```
<!-- Navbar Start -->
   <nav class="navbar navbar-expand-lg bg-white navbar-light
sticky-top px-4 px-lg-5 py-lg-0">
       <a href="index.html" class="navbar-brand d-flex align-
items-center">
           <h3 class="m-0"><i class="fa fa-building text-primary
me-3"></i>Аналітична веб-система побутового споживання
енергії</h3>
       </a>
       <button type="button" class="navbar-toggler" data-bs-
toggle="collapse" data-bs-target="#navbarCollapse">
           <span class="navbar-toggler-icon"></span>
       </button>
       <div class="collapse navbar-collapse" id="navbarCollapse">
           <div class="navbar-nav ms-auto py-3 py-lg-0">
               <a href="index.html" class="nav-item nav-link
active">Головна</a>
               <a href="about.html" class="nav-item nav-link">Про
нас</a>
      <div class="nav-item dropdown">
         <a href="service.html" class="nav-link dropdown-toggle"
data-bs-toggle="dropdown">Наші сервіси</a>
         <div class="dropdown-menu bg-light m-0">
            <a href="price.html" class="dropdown-item">Ціни на
тарифи</a>
            <a href="pokazi.html" class="dropdown-item">Внесення
показників</a>
            <a href="calc.html" class="dropdown-item">Калькулятор
розрахунку</a>
            <a href="tarif.html" class="dropdown-item">Діючі
тарифи</a>
            <a href="appointment.html" class="dropdown-
item">Заявка на підключення</a>
         </div>
      </div>
      <a href="support.html" class="nav-item nav-link">Зворотній
зв'язок</a>
           </div>
       </div>
   </nav>
   <!-- Navbar End -->
```

В наступному пункті, при програмуванні серверної частини вебсайту нам необхідно буде змінити частину верстки під вимоги фреймворка Django.

3.7 Програмування серверної частини вебсайту. Backend

Для програмування бекенду нашої веб-системи ми обрали фреймворк Django та мову Python. Встановлюємо Python [16] і також встановлюємо безкоштовну (Community) програму PyCharm [9], тому що вона працює як з python так і з Django [8]. Спочатку створимо проєкт в Pycharm.

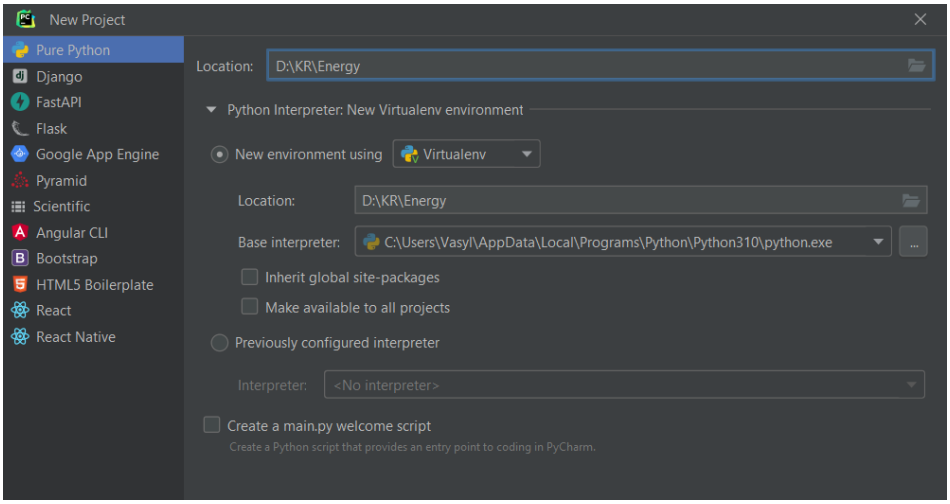


Рис. 3.22 - Створення проєкту в Pycharm

Далі створюємо віртуальне середовище, для того, щоб залежності встановлювалися локально в проєкт, а не глобально в операційну систему.

```
(Energy) PS D:\KR\Energy>  python -m venv venv
```

Рис. 3.23 - Створення віртуального середовища

Після цього, щоб увійти в віртуальне середовище потрібно активувати скрипт.

```
(Energy) PS D:\KR\Energy> D:\KR\Energy\venv\Scripts\Activate.ps1
```

Рис. 3.24 - Активація скрипту віртуального середовища

Після описаних дій ми опинимося в віртуальному середовищі.

```
(venv) PS D:\KR\Energy>
```

Рис. 3.25 - Активне віртуальне середовище

Далі по черзі запускаємо команди:

- pip install django – встановлюємо фреймворк Django
- django-admin startproject energy_system – створюємо проєкт сайта
- cd .\energy_system\ - переходимо в папку з проєктом
- python manage.py runserver – запускаємо сервер

Після чого переходимо за посиланням http://127.0.0.1:8000/ і бачимо стартову сторінку Django-проєкта.

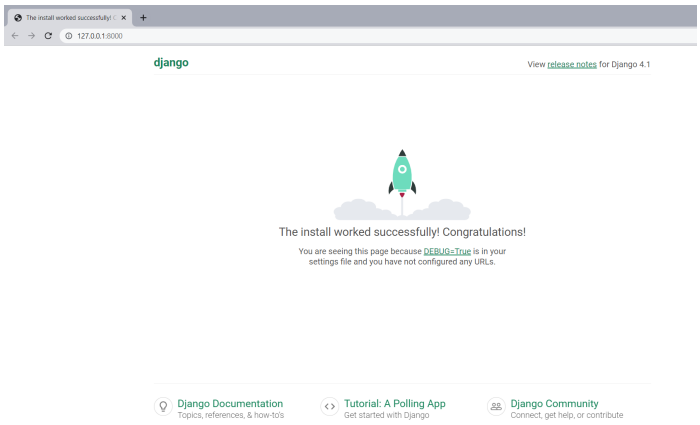


Рис. 3.26 - Стартова сторінка Django-проєкта

Для того, щоб зупинити сервер Django необхідно в терміналі натиснути сполучення клавіш CTRL+C.

Далі проводимо міграції:

- python manage.py migrate

Ця команда створить в базі даних вже готові по замовчуванню таблиці, наприклад таблиці для адмін-панелі.

У фреймворку Django admin-панель створюється разом із створенням проєкту. Доступна панель за посиланням http://127.0.0.1:8000/admin

Але перш ніж в адмін-панель увійти, потрібно створити за допомогою терміналу суперюзера:

- python manage.py createsuperuser

Username: admin

Email: admin@examples.com

Password: Vinnycya

Запускаємо сервер, вводимо створені логін та пароль і потрапляємо в адмін-панель:

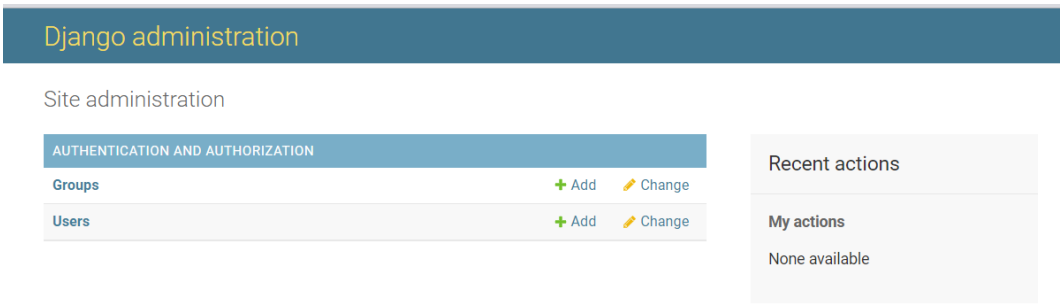


Рис. 3.27 - Адмін-панель Django

Для того, щоб змінити інтерфейс з англійської на українську мову потрібно в settings.py знайти LANGUAGE_CODE = 'en-en'

Змінити її на LANGUAGE_CODE = 'uk', і адмін-панель стає українською.

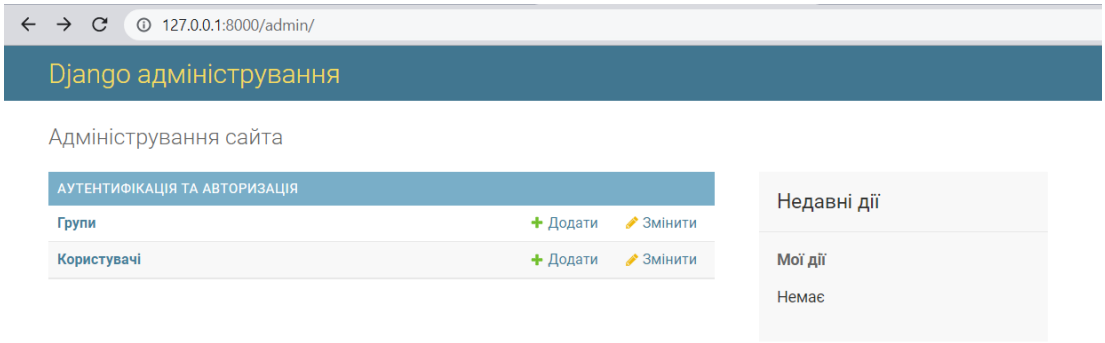


Рис. 3.28 - Українізована адмін-панель Django

Наступним кроком створюємо базовий додаток в проєкті django:

```
(venv) PS D:\KR\Energy\energy_system> python manage.py startapp bases
```

Рис. 3.29 Створення базового додатку в проєкті django

Реєструємо створений додаток в settings.py

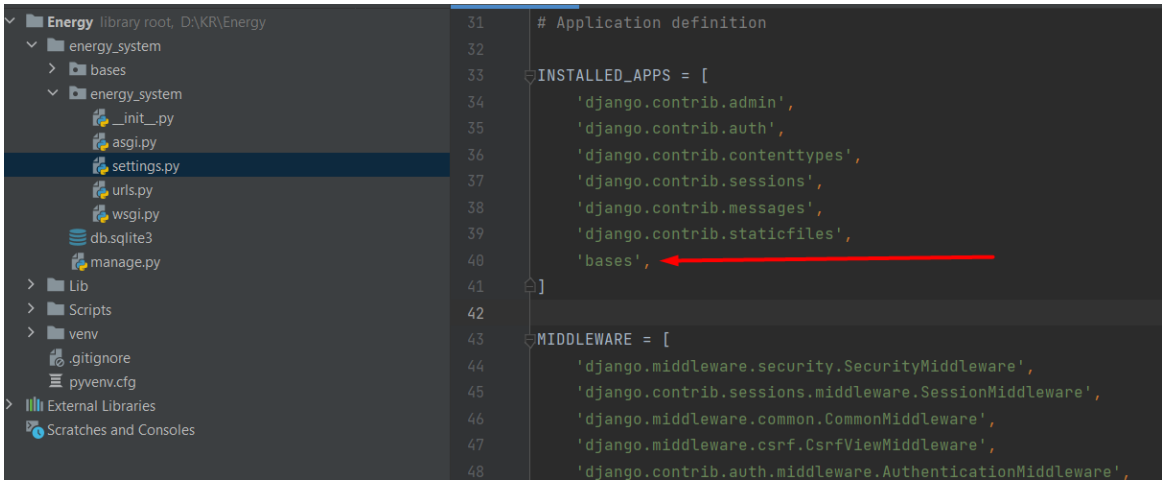


Рис. 3.30 - Реєстрація додатку bases в settings.py

Створюємо головну сторінку сайта:

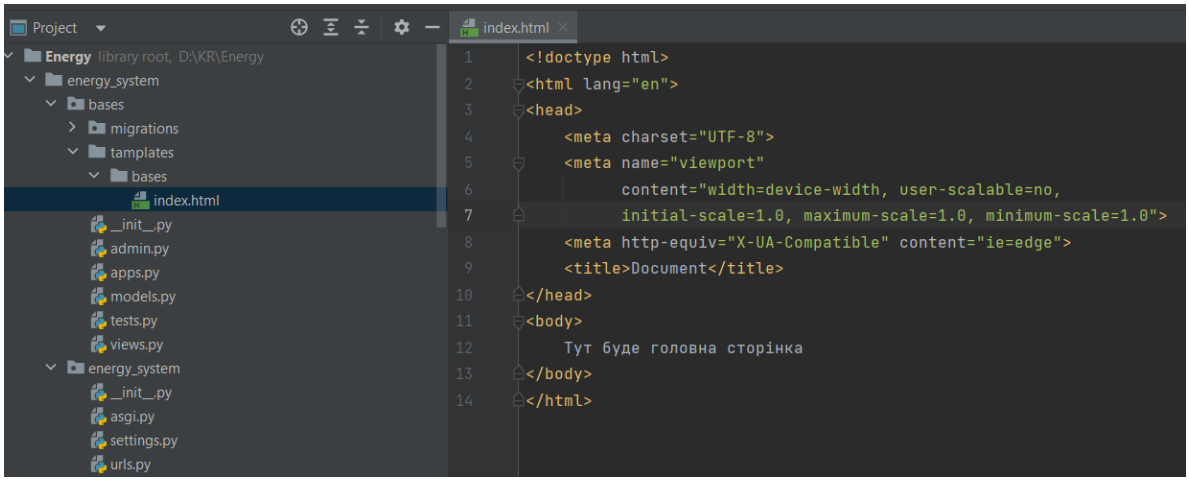


Рис. 3.31 - Створення головної сторінки сайта

І в представленні bases/views.py описуємо функцію яка буде викликати файл index.html

```
from django.shortcuts import render

def index(requests):
    return render(requests, 'bases/index.html')
```

Зв’язуємо в energy_system/urls.py створену функцію з url адресою:

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('bases.urls')),
]
```

Запускаємо сервер і бачимо що підключився файл index.html

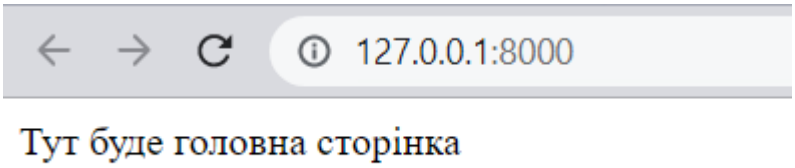


Рис. 3.32 - Запуск головної сторінки на сервері Django

Для того, щоб Django [28] коректно відображав зображення та підключав css стилі та js скрипти, необхідно провести деякі налаштування. Для цього в файлі setting.py необхідно дописати код:

```
STATIC_URL = 'static/'
STATICFILES_DIRS = [
    BASE_DIR / "static",
    "/var/www/static/",
]
```

В файли html потрібно буде дописувати код

{% load static %}

І потім, щоб вставити зображення, потрібно буде писати код наприклад ось так [25]:

```
<img src="{% static 'images/logo.png' %}">
```

Аналогічно підключимо файли CSS:

```
<link href="{% static 'css/style.css' %}" rel="stylesheet">
```

та JS:

```
<script src="{% static 'js/all.js' %}"></script>
```

Також в папці static створюємо папки css, lib, js та img і переносимо всі файли в них із файлів нашої верстки.

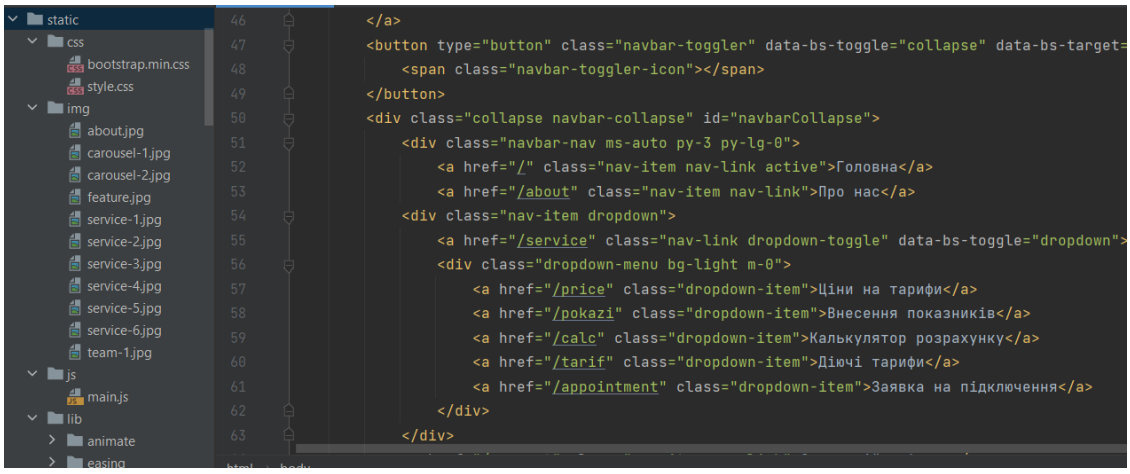


Рис. 3.33 - Архітектура папки static

Відредагований під вимоги Django index.html код сайту можна подивитися в додатку А.

В підсумку сайт виглядає так само як при верстці, але завдяки Django-серверу він функціонує так само, як сайт розміщений на хостингу.

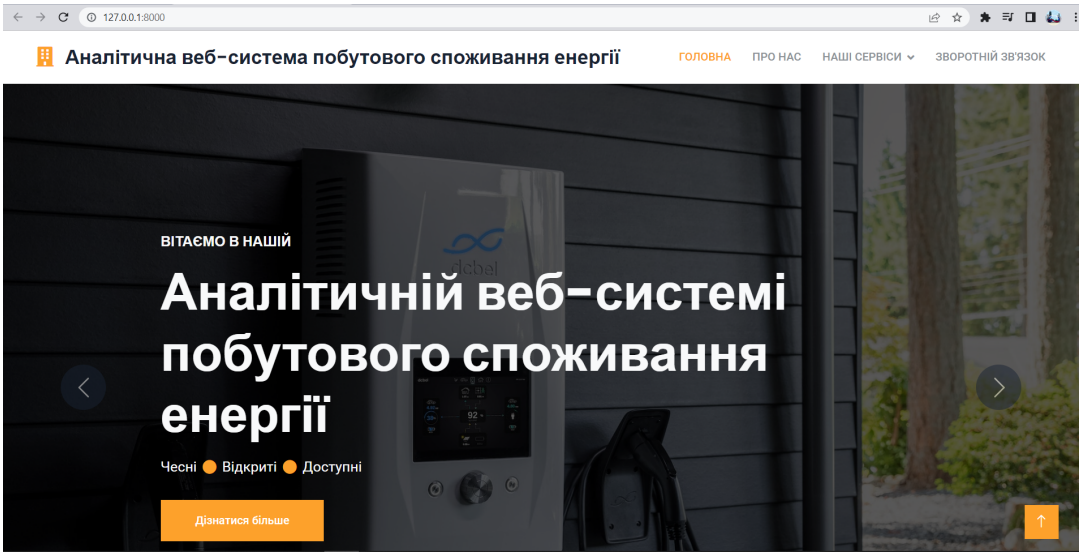


Рис. 3.34 - Вигляд головної сторінки на сервері Django

Далі створюю моделі (таблиці) в базі даних. Почнемо із моделі для сторінки price – цін на тарифи. Для цього в bases/models.py створимо клас Price,

```
from django.db import models

class Price(models.Model):
    name= models.TextField()
    one_class = models.CharField(max_length=30)
    two_class = models.CharField(max_length=30)

    @staticmethod
    def get_all_prices():
        return Price.objects.all()

    def __str__(self):
        return self.name
```

в якому створимо три поля з назвою споживачів та цінами для першого та другого класів.

Після чого створимо міграцію:

- python manage.py makemigrations

Та виконаємо її:

- python manage.py migrate

Для того, щоб клас відобразився в адмін-панелі реєструємо його в admin.py

```
from django.contrib import admin
from bases.models import Price

admin.site.register(Price)
```

Після чого наша модель відобразиться в панелі адміністратора

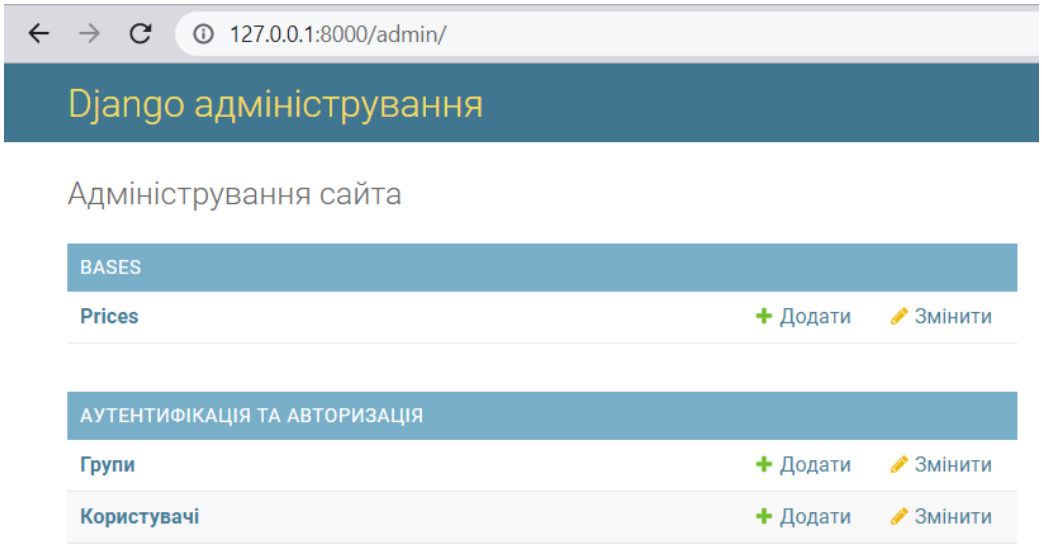


Рис. 3.35 - Відображення моделі Price в панелі адміністратора

Вносимо дані через адмін-панель

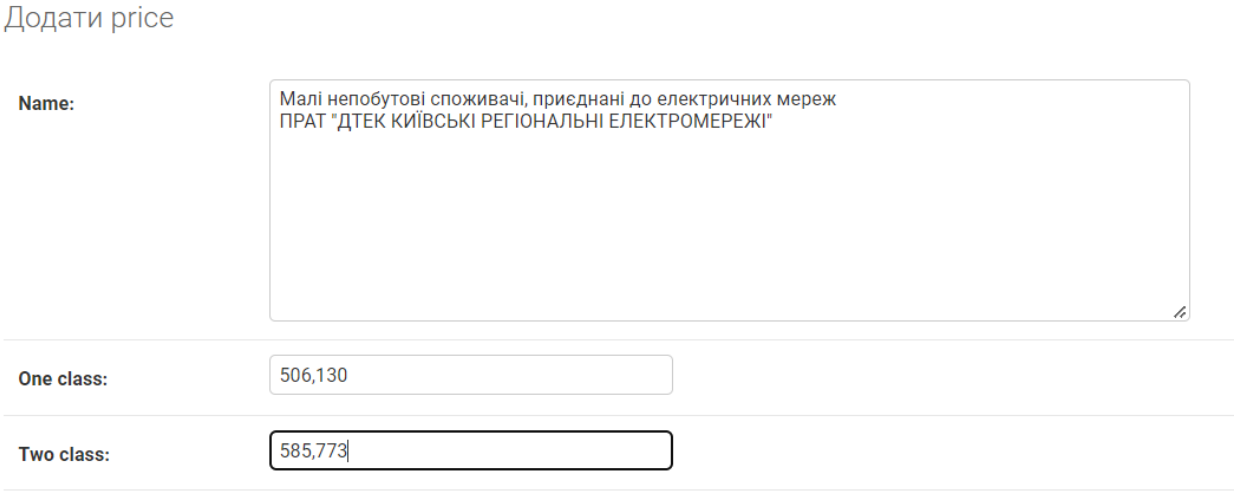


Рис. 3.36 - Додавання даних в модель Price в адмін-панелі

Далі в bases/views.py додаємо метод price

```
def price(request):
    price = Price.objects.all()
    return render(request, 'bases/price.html', {'price' : price})
```

і в файл price.html додаємо код

```
<table width="1200" class="table-bordered">
    <tr>
        <th>№</th>
        <th class="text-center">Споживачі</th>
        <th>1 клас</th>
        <th>2 клас</th>
    </tr>
        {% for el in price_data%}
            <tr>
                <td>{{ el.id }}</td>
                <td>{{ el.name }}</td>
                <td>{{ el.one_class }}</td>
                <td>{{ el.two_class }}</td>
            </tr>
        {% endfor %}
</table>
```

В результаті на сторінці з цінами на тарифи http://127.0.0.1:8000/price/ виведеться результат з бази даних.

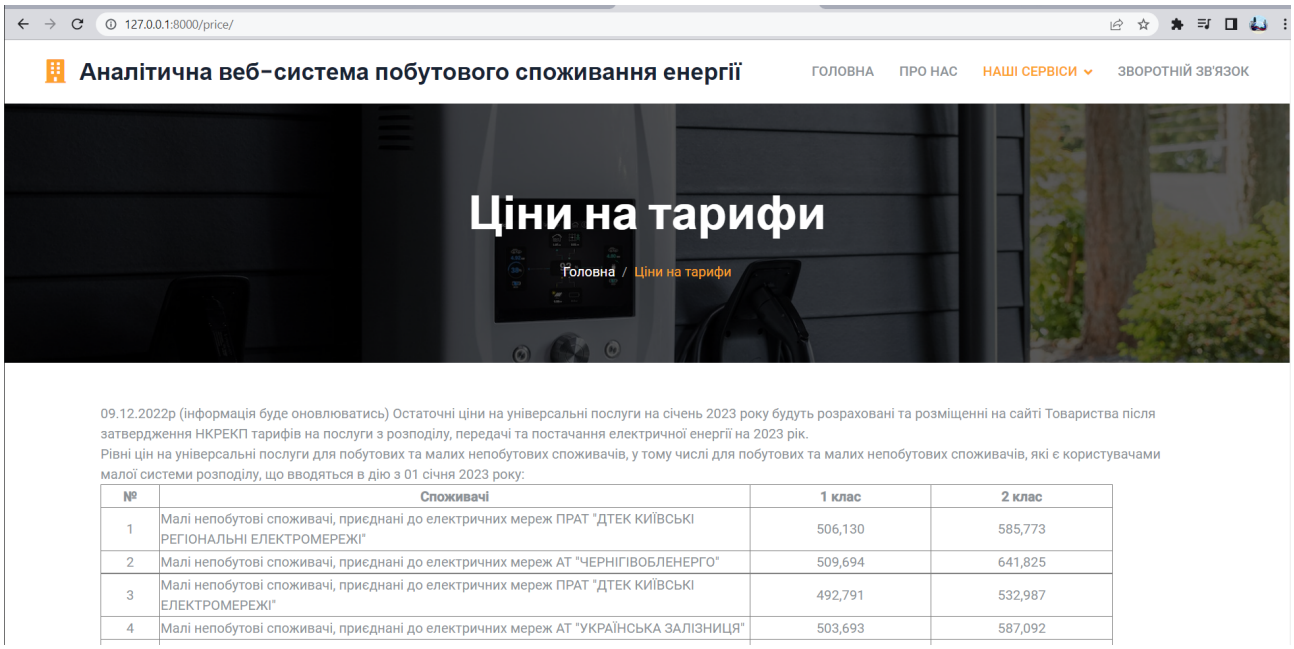


Рис. 3.37 - Вигляд сторінки з цінами на тарифи

До цього моменту ми розглядали приклад із введенням данних із адмін-панелі і виведенням цих даних на сторінці сайта. Далі розглянемо інший приклад із введенням даних із сторінок сайту, щоб дані записувалися в базу даних і відображалися в адмін-панелі. Зробимо це із сторінкою подачі заявки на підключення.

Для цього в models.py створимо клас Appointment

```
class Appointment(models.Model):
    first_name = models.CharField('Прізвище та ім\'я',
max_length=50)
    email = models.CharField('email', max_length=50)
    adresa = models.CharField('Ваша адреса', max_length=50)
    more_information = models.TextField('Вкажіть бажаний час або
ще якусь додаткова інформація')
    data_public = models.DateField('Дата подачі заявки')

    def __str__(self):
        return self.first_name
```

В bases створимо forms.py і заповнимо його даними для відображення нашої форми подачі заяки

```
from .models import Appointment
from django.forms import ModelForm, TextInput, Textarea, DateInput

class AppointmentForm(ModelForm):
    class Meta:
        model = Appointment
        fields = ['first_name', 'email', 'adresa',
'more_information', 'data_public']

        widgets = {
            'first_name': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Прізвище та ім\'я',
                'id': "gname",
            }),
            'email': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': "Email",
                'id': "gmail",
            }),
            'adresa': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Адреса підключення',
                'id': "adresa",
            }),
            'more_information': Textarea(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Додаткові відомості',
                'id': "message",
            }),
            'data_public': DateInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Бажана дата підключення у форматі
```

```
01.01.2023',
                'id': "date_appointment",
            }),

        }
```

В bases/views.py створюємо функцію appointment

```
def appointment(requests):
    error = ''
    if requests.method == 'POST':
        form = AppointmentForm(requests.POST)
        if form.is_valid():
            form.save()
        else:
            error = 'Форма заповнена неправильно'
    form = AppointmentForm()
    data = {
        'form': form,
        'error': error,
    }
    return render(requests, 'bases/appointment.html', data)
```

Для відображення введених даних в адмін-панелі додамо код в admin.py

```
from django.contrib import admin
from bases.models import Price, Appointment

admin.site.register(Price)

@admin.register(Appointment)
class AppointmentAdmin(admin.ModelAdmin):
    pass
```

Також змінемо appointment.html наступним чином

```
<div class="h-100 d-flex flex-column justify-content-center p-5">
    <h1 class="mb-4">Подати заявку</h1>
    <form method="post">
        {% csrf_token %}<br>
        {{form.first_name}}<br>
        {{form.email}}<br>
        {{form.adresa}}<br>
        {{form.more_information}}<br>
        {{form.data_public}}<br>
        <span>{{error}}</span>
        <div class="col-12">
            <button class="btn btn-primary w-100 py-3"
type="submit">Подати заявку</button>
        </div>
    </form>
</div>
```

Після створення та проведення міграцій отримаємо форму подачі заявки за адресою http://127.0.0.1:8000/appointment/

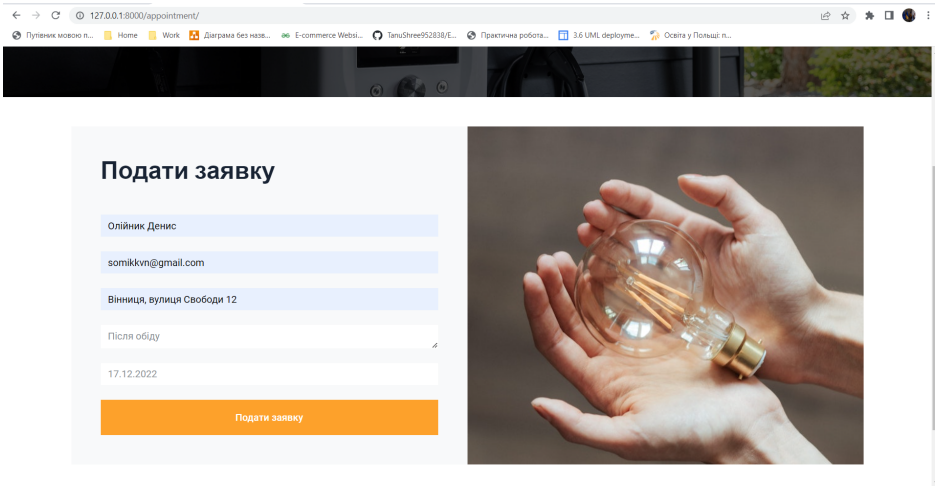


Рис. 3.38 - Вигляд сторінки «Подати заявку»

І після подачі заявки дані відобразяться в панелі адміністратора

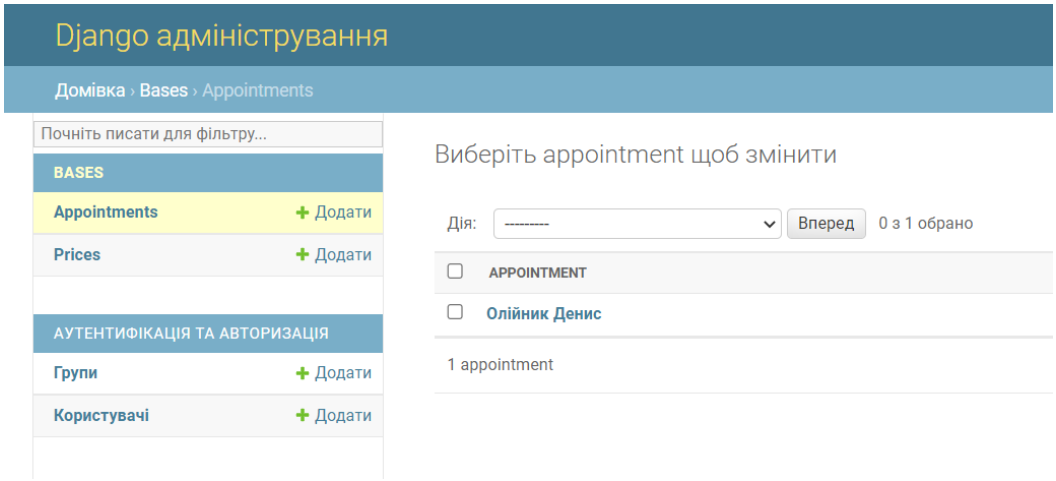


Рис. 3.39 - Відображення сторінки «Подати заявку» в панелі адміністратора

Всі інші сервіси створюються за аналогією розглянутих прикладів. Повний лістинг основних методів і класів можна переглянути в додатку Б.

3.8 Комерційний та технологічний аудит науково-технічної розробки

Для тестування верстки нашої веб-системи використаємо декілька інструментів. Спочатку перевіремо верстку нашого вебсайта за допомогою сервісу Markup Validation Service [33]. Розглянемо це на прикладі головної сторінки файлу index.html. Для цього перейдемо на сайт і обиремо вкладку

“Validate by File Upload” після чого завантажимо зазначений файл і натиснемо кнопку “Check”.

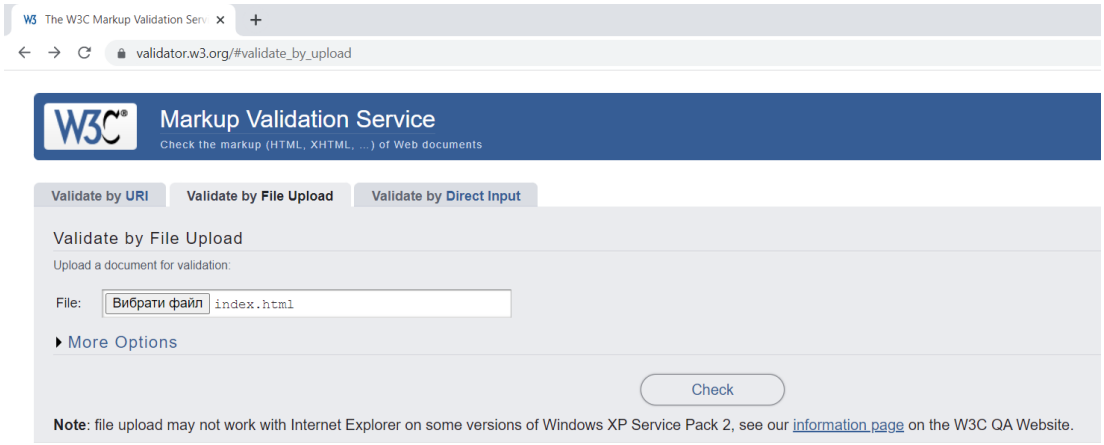


Рис. 3.40 - Перевірка верстки головної сторінки

В результаті отримуємо повідомлення «Перевірку документів завершено. Немає помилок або попереджень для показу».

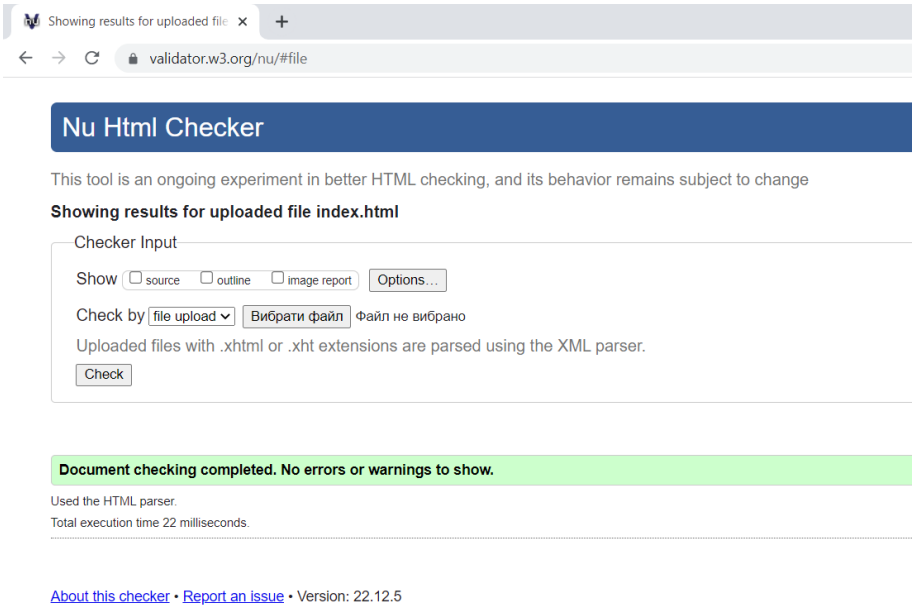


Рис. 3.41 - Результат перевірки верстки головної сторінки

Кожен вебсервіс ми перевірили і протестували в попередніх розділах, тому переходимо до тестування вебсайту в панелі розробника. Знову для прикладу покажемо це на прикладі головної сторінки.

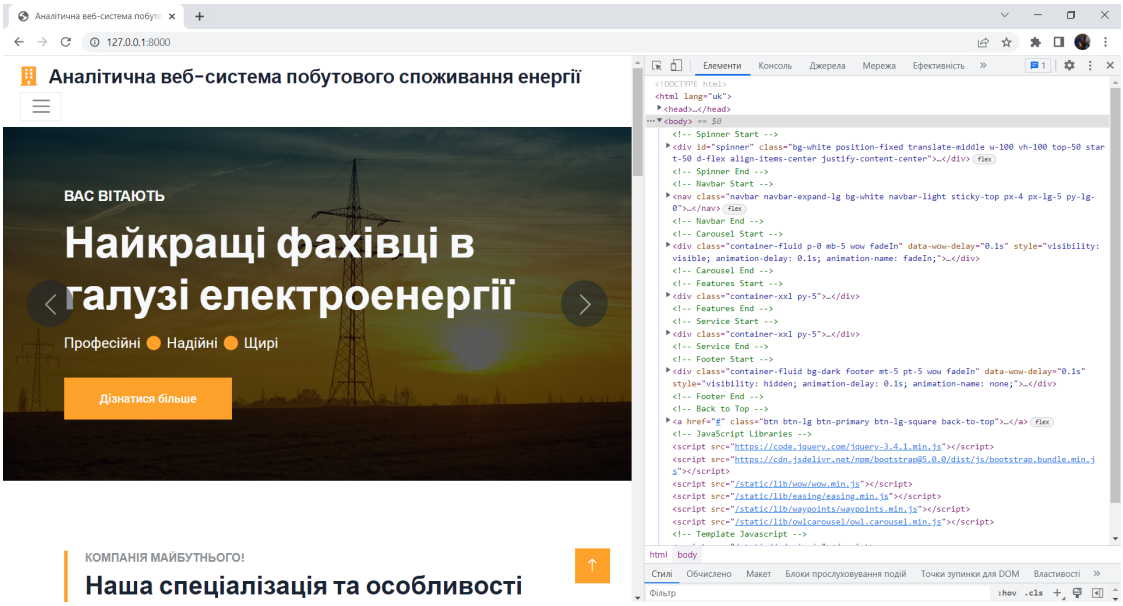


Рис. 3.42 - Результат перевірки на помилки в панелі розробника

Як бачимо, помилок знову не виявлено.

Перевіримо ще вебсайт на адаптивність, тобто його вигляд на різних мобільних пристроях. Спочатку перевіримо у режимі планшету.

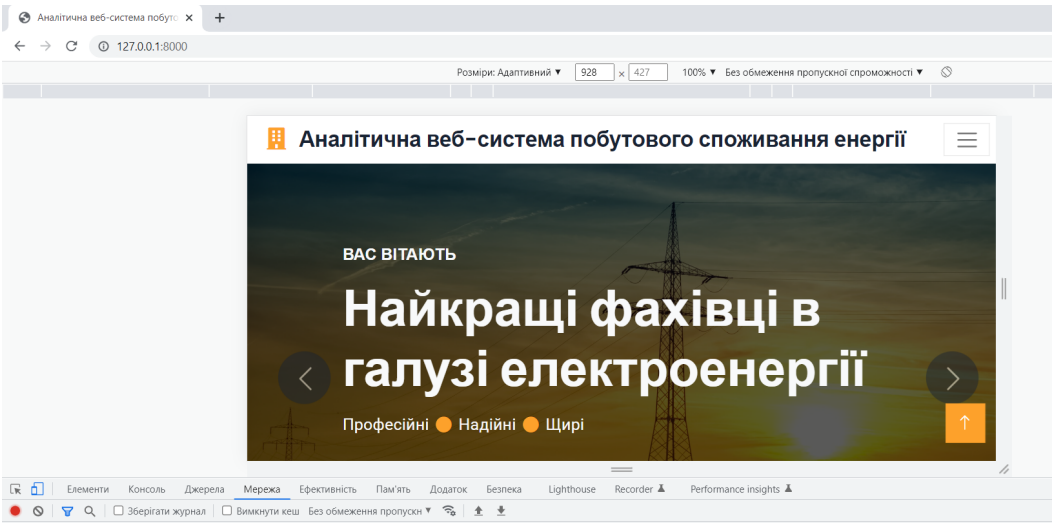


Рис. 3.43 - Результат перевірки вебсайту на адаптивність для планшету

Ще перевіримо для мобільного пристрою, наприклад, для iPhone 12 Pro.

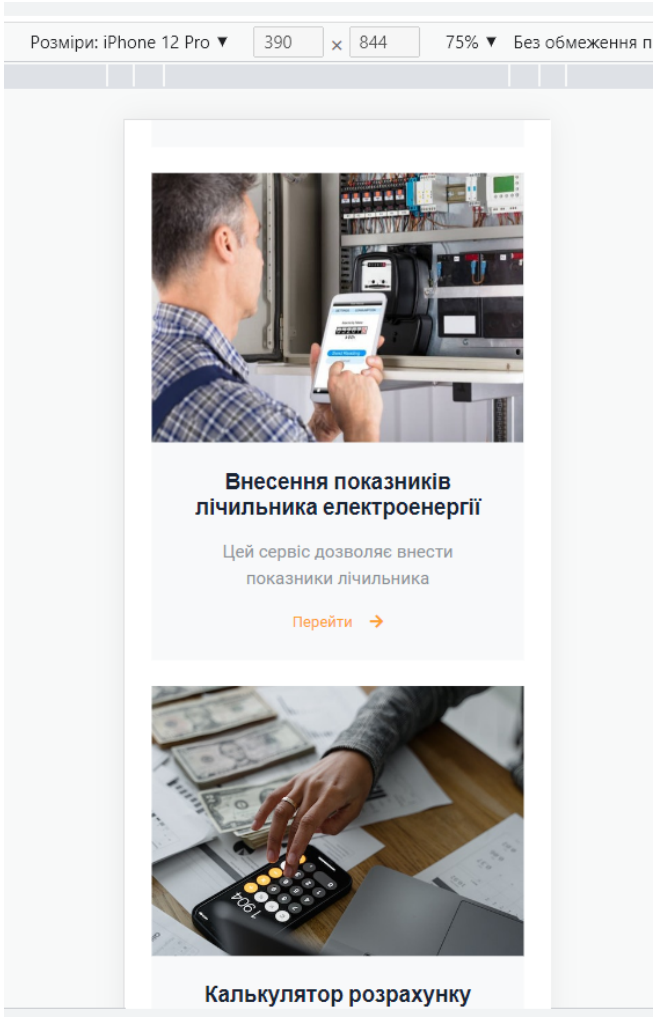


Рис. 3.44 - Результат перевірки вебсайту на адаптивність для iPhone 12 Pro

Знову впевнюємось, що сайт виглядає правильно. Отже, веб-система створена і перевірена, завантажуємо її на Github [15].

ВИСНОВКИ

Всі завдання, які необхідно виконати для початку використання аналітичної веб-системи побутового споживання енергії, виконані:

- огляд існуючих вебсистем
- огляд існуючих мов програмування
- огляд баз даних;
- огляд вебфреймворків
- розробка ER-діаграми
- розробка схеми класів UML
- вибір докального вебсервера для розгортання вебсистеми
- дизайн інтерфейсу користувача
- написання та тестування коду
- використання системи контролю версій.

Зазначимо, що після розгортання вебсайту на реальному хостингу, можна виконувати інші налаштування. Наприклад можна задіяти:

- Сервіс перевірки швидкості завантаження сайту
- Сервіс для перевірки навантаження на сайт
- Автоматичне кросбраузерне тестування сайта
- Сервіс для перевірки відкритих портів на будь-якому хостингу, по IP
- А також сервіси показу Ping, Traceroute, Whois.

Створену вебсистему можна використовувати в реальних проєктах, для інформування побутових споживачів електроенергії та аналітики даних.

РОЗДІЛ 4. ЕКОНОМІЧНА ЧАСТИНА

4.1 Комерційний та технологічний аудит науково-технічної розробки

Метою даного розділу є проведення технологічного аудиту, в даному випадку нового програмного продукту аналітичної веб-система побутового споживання енергії.

Особливістю програми є те, що дана технологія призначена для інтерактивного моніторингу побутового використання енергії на основі якого буде відбуватися коригування навантаження на електричні мережі для його зменшення.

Аналогом може бути Bissoft, ціна 1500 грн; Smart maic, ціна 920 грн.

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів. Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, у відповідності із табл. 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздат-ність продукту в реальних умовах
2	Багато аналогів на малому ринку	Ринкові п Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку

Продовження табл. 4.1

Ринкові переваги					
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною .	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практик на здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так із комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження табл. 4.1

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Усі дані по кожному параметру занесено в таблиці 4.2

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії оцінювання	ПІБ експертів		
	Експерт 1	Експерт 2	Експерт 3
	Бали		
Технічна здійсненність концепції	3	3	4
Наявність аналогів на ринку	4	3	4
Цінова політика	4	4	3
Технічні та споживчі властивості виробу	4	3	4
Експлуатаційні витрати	3	4	3
Ринок збуту	4	3	4
Конкурентоспроможність	3	4	3
Фахівці з технічної і комерційної реалізації	4	3	4
Фінансування	4	4	3
Матеріально-технічна база	3	4	4
Термін реалізації ідеї	4	4	4
Супровідна документація	3	3	4
Сума	43	42	44
Середньоарифметична сума балів	(43+42+44) / 3 = 43		

За даними таблиці 4.2 можна зробити висновок щодо рівня комерційного потенціалу даної розробки. Для цього доцільно скористатись рекомендаціями, наведеними в таблиці 4.3.

Таблиця 4.3 - Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 - 1 0	Низький
1 1 - 2 0	Нижче середнього
2 1 - 3 0	Середній
3 1 - 4 0	Вище середнього
4 1 - 4 8	Високий

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є високим, що досягається за рахунок того, що програмний продукт відрізняється від існуючих тим, що дана технологія призначена для інтерактивного моніторингу побутового використання енергії на основі якого буде відбуватися коригування навантаження на електричні мережі для його зменшення.

4.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

4.2.1 Основна заробітна плата розробників, яка розраховується за формулою:

$$З_о = \frac{M}{T_р} \cdot t, \tag{4.1}$$

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

$T_р$ – число робочих днів в місяці, 20 днів;

t – число днів роботи розробника (дослідника).

Результати розрахунків зведемо до таблиці 4.1.

Таблиця 4.1 – Основна заробітна плата розробників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник проекту	44000	2200,00	35	77000,000
Програміст	40000	2000,00	35	70000,000
Всього				147000,00

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

4.2.2 Додаткова заробітна плата розробників, які приймали участь в розробці обладнання.

Додаткова заробітна плата прийнято розраховувати як 14,5 % від основної заробітної плати розробників та робітників:

$$З_д = З_о \cdot 14{,}5\ \% / 100\ \% \qquad (4.2)$$

$$З_д = (147000{,}00 \cdot 14{,}5\ \% / 100\ \%) = 21315{,}00 \ \text{(грн.)}$$

4.2.3 Нарахування на заробітну плату розробників.

Згідно діючого законодавства нарахування на заробітну плату складають 22 % від суми основної та додаткової заробітної плати.

$$Н_з = (З_о + З_д) \cdot 22\ \% / 100\% \qquad (4.3)$$

$Н_з = (147000,00 + 21315,00) \cdot 22\ \% / 100\ \% = 37029,30$ (грн.)

4.2.4. Оскільки для розроблювального пристрою не потрібно витрачати матеріали та комплектуючи, то витрати на матеріали і комплектуючі дорівнюють нулю.

4.2.5 Амортизація обладнання, яке використовувалось для проведення розробки.

Амортизація обладнання, що використовувалось для розробки в спрощеному вигляді амортизація обладнання, що використовувалась для розробки розраховується за формулою:

$$A = \frac{Ц}{Тв} \cdot \frac{t_{вик}}{12} \text{ [грн.].} \quad (4.4)$$

де Ц – балансова вартість обладнання, грн.;

Т – термін корисного використання обладнання згідно податкового законодавства, років

$t_{вик}$ – термін використання під час розробки, місяців

Розрахуємо, для прикладу, амортизаційні витрати на комп'ютер балансова вартість якого становить 20000 грн., термін його корисного використання згідно податкового законодавства – 2 роки, а термін його фактичного використання – 1,75 міс.

$$A_{обл} = \frac{20000}{2} \times \frac{1,75}{12} = 1458,33 \ грн.$$

Аналогічно визначаємо амортизаційні витрати на інше обладнання та приміщення. Розрахунки заносимо до таблиці 4.2. Для розрахунку амортизації нематеріальних ресурсів використовується формула:

$$A_{н.р.} = Ц_{н.р.} * Н_а * \frac{t_{вик.}}{12} \quad (4.5)$$

Але, так як вартість ліцензійної ОС та спеціалізованих ліцензійних нематеріальних ресурсів менше 20000 грн, то даний нематеріальний актив не амортизується, а його вартість включається у вартість розробки повністю, $B_{нем.ак.}$ = 1390 грн.

Таблиця 4.2 – Амортизаційні відрахування матеріальних і нематеріальних ресурсів для розробників

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортиза-ційні відрахуван-ня, грн.
Комп'ютер та комп'ютерна переферія	20000	2	1,75	1458,333
Офісне обладнення	25000	4	1,75	911,458
Приміщення	850000	20	1,75	6197,917
Всього				8567,71

5.2.6 Тарифи на електроенергію для непобутових споживачів (промислових підприємств) відрізняються від тарифів на електроенергію для населення. При цьому тарифи на розподіл електроенергії у різних постачальників (енергорозподільних компаній), будуть різними. Крім того, розмір тарифу залежить від класу напруги (1-й або 2-й клас). Тарифи на розподіл електроенергії для всіх енергорозподільних компаній встановлює Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП). Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п, \quad (4.6)$$

де В – вартість 1 кВт-години електроенергії для 1 класу підприємства, В = 6,2 грн./кВт;

П – встановлена потужність обладнання, кВт. П = 0,6 кВт;

Ф – фактична кількість годин роботи обладнання, годин.

$К_п$ – коефіцієнт використання потужності, $К_п$ = 0,9.

$В_е$ = 0,9 · 0,6 · 8 · 35 · 6,2 = 937,44 (грн.)

5.2.7 Інші витрати та загальновиробничі витрати.

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками. Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.7)$$

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

$I_в$ = 147000,00 * 75% / 100% = 110250 (грн.)

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін. Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$Н_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.8)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$Н_{нзв}$ = 147000,00 * 140 % / 100 % = 205800 (грн.)

5.2.9 Витрати на проведення науково-дослідної роботи.

Сума всіх попередніх статей витрат дає загальні витрати на проведення науково-дослідної роботи:

$В_{заг}$ = 147000,00+21315,00+37029,30+8567,71+1390+937,44+110250+
+205800 = 532289,45 грн.

5.2.11 Розрахунок загальних витрат на науково-дослідну (науково-технічну) роботу та оформлення її результатів.

Загальні витрати на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються *ЗВ*, визначається за формулою:

$$ЗВ = \frac{В_{заг}}{\eta} \text{ (грн)}, \qquad (5.9)$$

де η – коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи.

Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η=0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9. Оберемо η = 0,5, так як розробка, на даний момент, знаходиться на стадії дослідного зразка:

ЗВ = 532289,45 / 0,5 = 1064579 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку. Саме зростання чистого прибутку забезпечить потенційному інвестору надходження додаткових коштів, дозволить покращити фінансові результати його діяльності, підвищить конкурентоспроможність та може позитивно вплинути на ухвалення рішення щодо комерціалізації цієї розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

а) вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

б) зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 3-х років після її впровадження);

в) кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб'єктів (зацікавлених осіб) цього попиту;

г) визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов'язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу. При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

- абсолютного економічного ефекту (чистого дисконтованого доходу);

- внутрішньої економічної дохідності (внутрішньої норми дохідності);
- терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об'єднати, враховуючи визначені ситуації з відповідними умовами.

4.3.1 Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (4.10)$$

де $\pm\Delta Ц_o$ – зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки;

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

Цб – вартість програмного продукту у році до впровадження результатів розробки;

ΔN – збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

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

p – коефіцієнт, який враховує рентабельність продукту;

ϑ – ставка податку на прибуток, у 2022 році ϑ =18%.

Припустимо, що при прогнозованій ціні 500 грн. за одиницю виробу, термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти її ціну на *50* грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року – на 13000 шт., протягом другого року – на 23000 шт., протягом третього року на 33000 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

*$\Delta П_1$ = (0*50 + (500 + 50)*13000)* 0,8333* 0,43) * (1 - 0,18) = 1909916,590 грн.*

*$\Delta П_2$ = (0*50 + (500 + 50)*(13000+23000)* 0,8333* 0,43) * (1 - 0,18) = 5817899,767 грн.*

*$\Delta П_3$ = (0*50 + (500 + 50)*(13000+23000+33000)* 0,8333* 0,43) * (1 - 0,18) = 11150974,554 грн.*

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 18878790,91 грн.

4.3.2 Розрахунок ефективності вкладених інвестицій та періоду їх окупності.

Розраховуємо приведену вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної (науково-технічної) роботи, грн;

Т – період часу, протягом якою виявляються результати впровадженої науково-дослідної (науково-технічної) роботи, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, $\tau = 0,05...0,15$;

t – період часу (в роках).

Збільшення прибутку ми отримаємо починаючи з першого року:

ПП= $(1909916,590/(1+0,1)^1)+(5817899,767/(1+0,1)^2)+(11150974,554/(1+0,1)^3)$ = 1736287,81 + 4808181,626 + 8377892,227 = 14922361,66 грн.

Далі розраховують величину початкових інвестицій *PV*, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} * ЗВ, \quad (4.12)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}=2...5$, але може бути і більшим;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$PV = 2 * 1064579 = 2129157,79$ *грн.*

Тоді абсолютний економічний ефект $E_{абс}$ або чистий приведений дохід *(NPV, Net Present Value)* для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП – PV, \quad (4.13)$$

$E_{абс}$ = 14922361,66 – 2129157,79 = 12793203,87 грн.

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботи може бути доцільним.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність або показник внутрішньої но-рми дохідності *(IRR, Internal Rate of Return)* вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{4.14}$$

$T_ж$ – життєвий цикл наукової розробки, роки.

$$E_в = \sqrt[3]{(1 + 12793203{,}87/2129157{,}79} - 1 = 0{,}914$$

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

$$\tau = d + f, \tag{4.15}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022 році в Україні d = (0,09…0,14);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{\min} = 0{,}14 + 0{,}05 = 0{,}19.$$

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

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

$$T_{ок} = \frac{1}{E_в}, \tag{4.16}$$

$T_{ок}$ = 1 / 0,914 = 1,09 р.

Оскільки $T_{ок}$ < 3-х років, а саме термін окупності рівний 1,09 роки, то фінансування даної наукової розробки є доцільним.

Висновки до розділу: економічна частина даної роботи містить розрахунок витрат на розробку нового програмного продукту, сума яких складає 1064579 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення, розраховано період окупності витрат для інвестора та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є висококонкурентоспроможним. Період окупності складе близько 1,09 роки.

СПИСОК ЛІТЕРАТУРИ

1. ТОВ "Київська обласна енергопостачальна компанія" [Електронний ресурс]. – Режим доступу: https://koec.com.ua/
2. АТ "Полтаваобленерго" [Електронний ресурс]. – Режим доступу: https://www.poe.pl.ua/
3. Калькулятор розрахунку оплати за електроенергію [Електронний ресурс]. – Режим доступу: https://dom.ria.com/uk/kalkulyatory-zkh/kalkulyator-tseni-elektroenergii/
4. Діючі тарифи для населення ТОВ – Полтаваенергозбут [Електронний ресурс]. – Режим доступу: https://www.energo.pl.ua/
5. ПраТ «Закарпаттяобленерго» [Електронний ресурс]. – Режим доступу: https://zakarpat.energy/
6. Google відгуки про ПраТ«Закарпаттяобленерго» [Електронний ресурс]. – Режим доступу: https://www.google.com/search?q=Закарпаттяобленерго
7. Programming, scripting, and markup languages [Electronic resource] – https://insights.stackoverflow.com/survey/2021#key-territories-country
8. Django Підручник [Електронний ресурс] – Режим доступу до ресурса: https://w3schoolsua.github.io/django/index.html
9. IDE для професійної розробки на Python [Електронний ресурс] – Режим доступу до ресурса: https://www.jetbrains.com/pycharm/
10. JavaScript [Електронний ресурс] – Режим доступу до ресурса: https://developer.mozilla.org/en-US/docs/Web/JavaScript
11. Photoshop - Довідка й навчальні посібники [Електронний ресурс] – Режим доступу до ресурса: https://helpx.adobe.com/ua/pdf/photoshop_reference.pdf
12. The Python Tutorial [Електронний ресурс] – Режим доступу до ресурса: https://docs.python.org/uk/3/tutorial/index.html.
13. Query бібліотека JavaScript [Електронний ресурс] – Режим доступу до ресурса: https://jquery.com/

14. Query Підручник [Електронний ресурс] – Режим доступу до ресурса: https://w3schoolsua.github.io/jquery/index.html#gsc.tab=0
15. Вебсервіс для спільної розробки програмного забезпечення [Електронний ресурс] – Режим доступу до ресурса: https://github.com/
16. Завантажити Python. Офіційний сайт [Електронний ресурс] – Режим доступу до ресурса: https://www.python.org/
17. Інформатика (рівень стандарту): підручник для 10 (11) класів закладів загальної середньої освіти України / Й.Я. Ривкінд, Т.І. Лисенко, Л.А. Чернікова, В.В. Шакотько.- Київ: “Генеза”, 2018.- 144 с.
18. Кольори в HTML [Електронний ресурс] – Режим доступу до ресурса: https://html-css.co.ua/html/color-in-html/
19. Мова розмітки гіпертексту HTML [Електронний ресурс] – Режим доступу до ресурса: https://uk.wikipedia.org/wiki/HTML
20. Навчальні матеріали: Python [Електронний ресурс] – Режим доступу до ресурса: http://www.matfiz.univ.kiev.ua/pages/13.
21. Основи CSS [Електронний ресурс] – Режим доступу до ресурса: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics
22. SQL Підручник [Електронний ресурс] – Режим доступу до ресурса: https://w3schoolsua.github.io/sql/index.html#gsc.tab=0
23. Основні відомості про шари Photoshop [Електронний ресурс] – Режим доступу до ресурса: https://helpx.adobe.com/ua/photoshop/using/layer-basics.html
24. Підручник з HTML та CSS [Електронний ресурс] – Режим доступу до ресурса: https://htmlbook.at.ua/news/tutorial_css/1-0-2
25. Плугар І.В. Python з нуля [Електронний ресурс] – Режим доступу до ресурса: https://www.youtube.com/c/IoanPlugar_inf
26. Путівник мовою програмування Python. [Електронний ресурс] – Режим доступу до ресурса: https://pythonguide.rozh2sch.org.ua/

27. Фреймворк Bootstrap для створення адаптивних сайтів [Електронний ресурс] – Режим доступу до ресурса: https://getbootstrap.com/docs/5.0/getting-started/introduction/
28. Фреймворк Django [Електронний ресурс] – Режим доступу до ресурса: https://www.djangoproject.com/
29. Вікіпедія [Електронний ресурс] – Режим доступу до ресурса: https://uk.wikipedia.org/wiki/модель_сутність-зв'язок
30. Елементи UML [Електронний ресурс] – Режим доступу до ресурса: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-elements.html
31. Автоматизація прототипування інформаційних систем [Електронний ресурс] – Режим доступу до ресурса: https://posibniki.com.ua/post-avtomatizaciya-prototipuvannya--informaciynih-sistem
32. Що таке архітектура веб-сайтів? [Електронний ресурс] – Режим доступу до ресурса: https://uk.theastrologypage.com/website-architecture
33. Markup Validation Service [Електронний ресурс] – Режим доступу https://validator.w3.org/
34. PHP: Hypertext Preprocessor [Електронний ресурс] – Режим доступу https://www.php.net/
35. Розділ мови PHP [Електронний ресурс] – Режим доступу https://metanit.com/php/
36. MySQL [Електронний ресурс] – Режим доступу https://www.mysql.com/
37. MySQL Матеріал з Вікіпедії — вільної енциклопедії [Електронний ресурс] – Режим доступу https://uk.wikipedia.org/wiki/MySQL
38. PostgreSQL [Електронний ресурс] – Режим доступу https://www.postgresql.org/
39. What is PostgreSQL? [Електронний ресурс] – Режим доступу https://aws.amazon.com/rds/postgresql/what-is-postgresql/
40. SQLite [Електронний ресурс] – Режим доступу https://www.sqlite.org/index.html

41. SQLite tutorial [Електронний ресурс] – Режим доступу https://www.sqlitetutorial.net/
42. MongoDB: The Developer Data Platform [Електронний ресурс] – Режим доступу https://www.mongodb.com/home
43. MongoDB Tutorial [Електронний ресурс] – Режим доступу https://www.w3schools.com/mongodb/
44. SQL Server у локальному середовищі [Електронний ресурс] – Режим доступу https://www.microsoft.com/ru-ru/sql-server/sql-server-downloads
45. Introducing Microsoft SQL Server 2019 [Електронний ресурс] – Режим доступу https://clouddamcdnprodep.azureedge.net/gdc/gdcJivzXl/original
46. CherryPy — A Minimalist Python Web Framework [Електронний ресурс] – Режим доступу https://docs.cherrypy.dev/en/latest/
47. Flask's documentation [Електронний ресурс] – Режим доступу https://flask.palletsprojects.com/en/2.2.x/
48. Welcome to Pyramid, a Python Web Framework [Електронний ресурс] – Режим доступу https://trypyramid.com/
49. Web2py Web Framework [Електронний ресурс] – Режим доступу http://www.web2py.com/
50. Use Chrome Developer Tools to check tags [Електронний ресурс] – Режим доступу https://support.google.com/campaignmanager/answer/2828688?hl=en

Додаток А
(обов'язковий)

ВНТУ

ЗАТВЕРДЖЕНО
Зав. кафедри КСУ ВНТУ,
д.т.н., доцент

В'ячеслав КОВТУН
"03" жовтня 2022 року

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

Аналітична веб-система побутового споживання енергії

08-33.МКР.08.00.000 ТЗ

Студент групи 2АКІТ-21м

Підпис

Саранчук Сергій
Ім'я ПРІЗВИЩЕ

Керівник

Підпис

Володимир Дубовой
Ім'я ПРІЗВИЩЕ

Вінниця 2022

1. Назва та галузь застосування

1.1. Назва – Аналітична веб-система побутового споживання енергії

1.2. Галузь застосування – споживачі/постачальники електроенергії.

2. Підстава для проведення розробки.

Тема магістерської кваліфікаційної роботи затверджена наказом по ВНТУ від “14” вересня 2022 року №203

3. Мета та призначення розробки.

Метою магістерської кваліфікаційної роботи є покращення процесу розрухунку вартості та подачі показнців спожитої електроенергії.

4. Джерела розробки.

Магістерська кваліфікаційна робота виконується вперше. В ході проведення розробки повинні використовуватись такі документи:

1. Django Підручник [Електронний ресурс] – Режим доступу до ресурса: https://w3schoolsua.github.io/django/index.html

2. Автоматизація прототипування інформаційних систем [Електронний ресурс] – Режим доступу до ресурса: https://posibniki.com.ua/post-avtomatizaciya-prototipuvannya--informaciynih-sistem5. Вимоги до розробки.

5.1. Перелік головних функцій:

- авторизація користувачів;
- оптимізація процесів розрхунку вартості та подачі показників спожитої електроенергії;
- керування базою даних;

5.2. Основні технічні вимоги до розробки.

5.2.1. Вимоги до програмної платформи:

- WINDOWS 10;
- Хмарне середовище Azure;
- Visual Studio 2019.

5.2.2. Умови експлуатації системи:

- робота на мобільних та веб-додатках;
- можливість цілодобового функціонування системи;
- дані оновлюються і є актуальними.

6. Стадії та етапи розробки.

6.1 Пояснювальна записка:

1. Аналіз методів, принципів, підходів і засобів реалізації задачі автоматизації процесами в об’єкті управління відповідно до теми кваліфікаційної роботи. Постановка задач дослідження «03» 09 2022 р.
2. Визначення технічних характеристик системи «18» 09 2022 р.
3. Розробка програмного забезпечення системи «02» 10 2022 р.

6.2 Графічні матеріали:

1. Розробка UML-діаграм системи «16» 10 2022 р.
2. Розробка моделі бази даних системи «20» 11 2022 р.
3. Тестування програмного забезпечення «25» 11 2022 р.

7. Порядок контролю і приймання.

7.1. Хід виконання роботи контролюється керівником роботи. Рубіжний контроль провести до «28» 11 2022 р.

7.2. Атестація МКР здійснюється на попередньому захисті. Попередній захист магістерської кваліфікаційної роботи провести до «16» 12 2022 р.

7.3. Підсумкове рішення щодо оцінки якості виконання роботи приймається на засіданні ЕК. Захист магістерської кваліфікаційної роботи провести до «23» 12 2022 р.

Додаток Б

(обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

Аналітична веб система побутового споживання енергії

Студент групи 2АКІТ-21

Саранчук Сергій

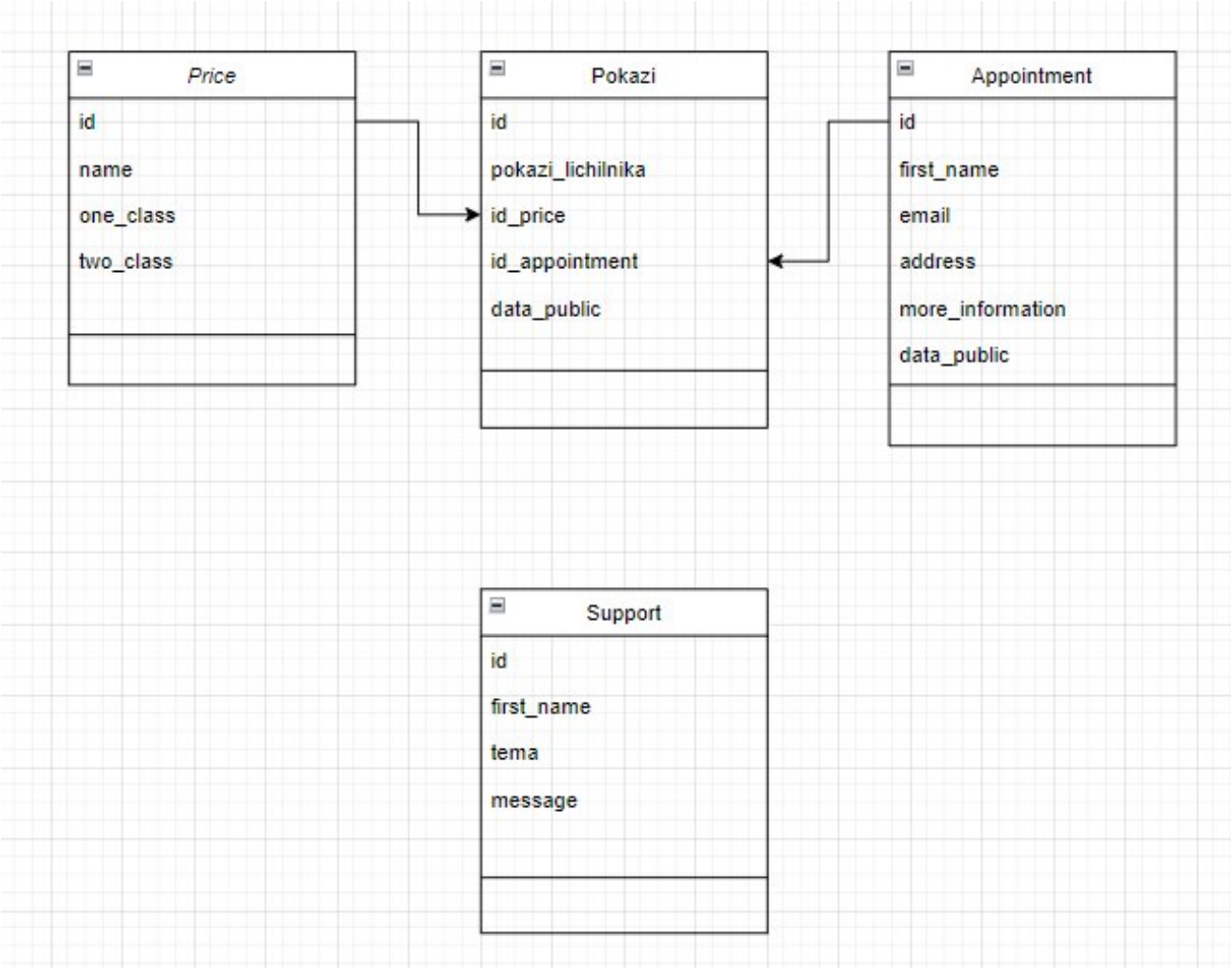
Підпис Ім'я ПРІЗВИЩЕ

Керівник д.т.н., професор, професор каф. КСУ

Володимир Дубовой

Підпис Ім'я ПРІЗВИЩЕ

ER – модель


Price
id
name
one_class
two_class
Pokazi
id
pokazi_lichilnika
id_price
id_appointment
data_public
Appointment
id
first_name
email
address
more_information
data_public
Support
id
first_name
tema
message

UML - діаграма класів

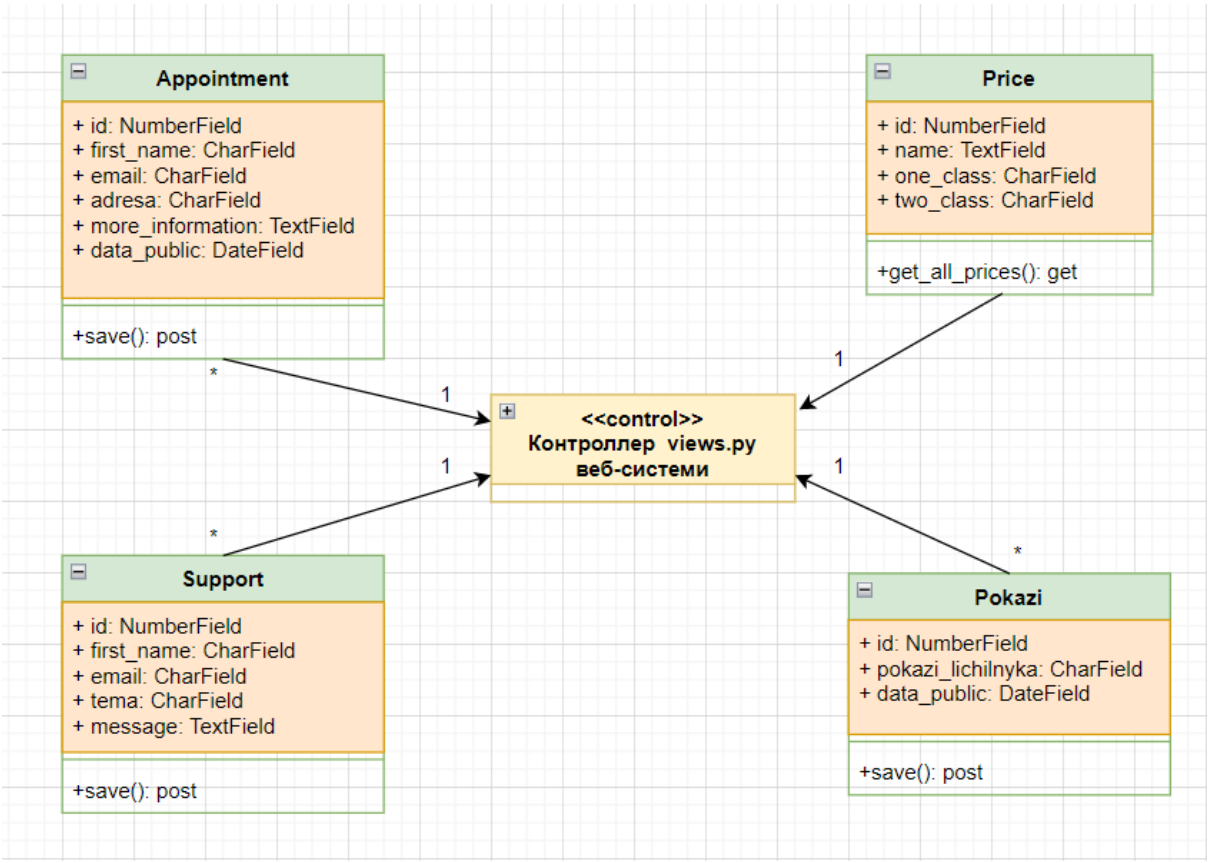


Схема роботи застосунку з server side rendering

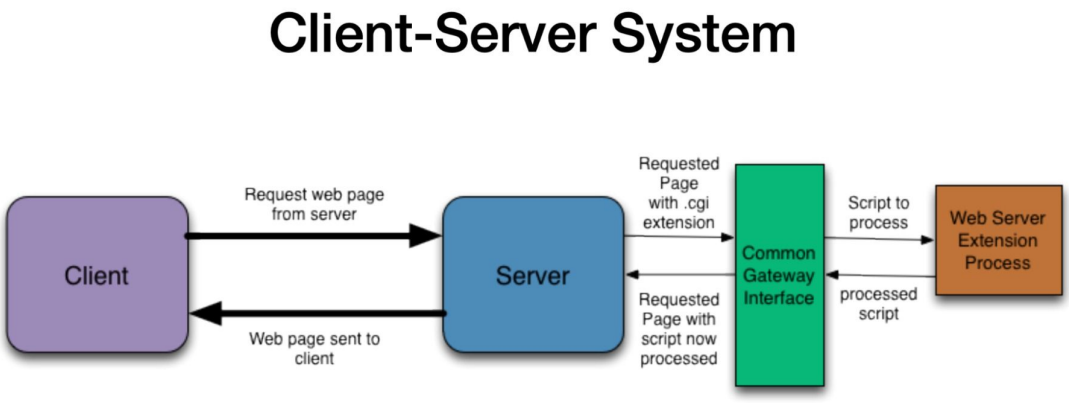
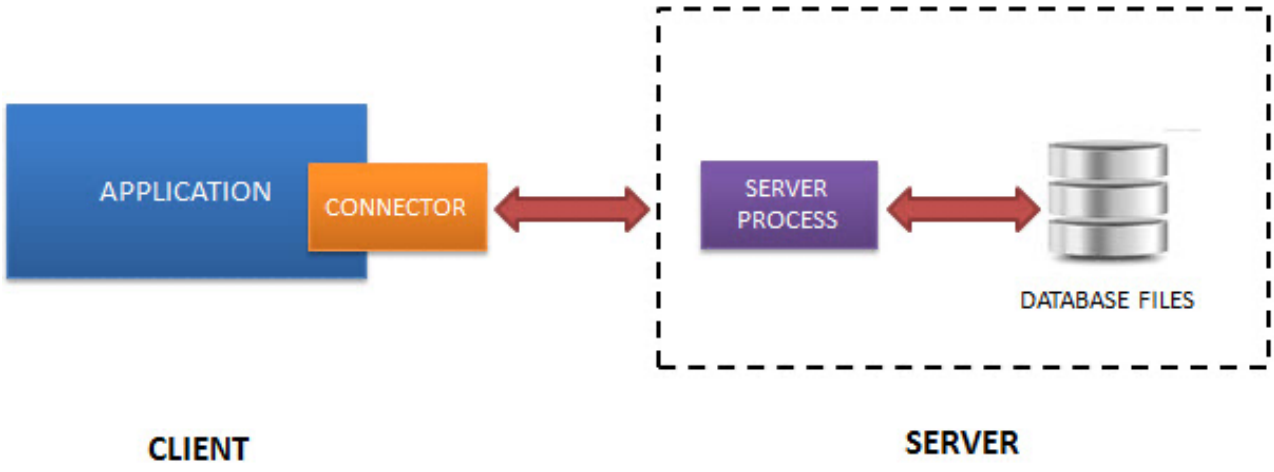
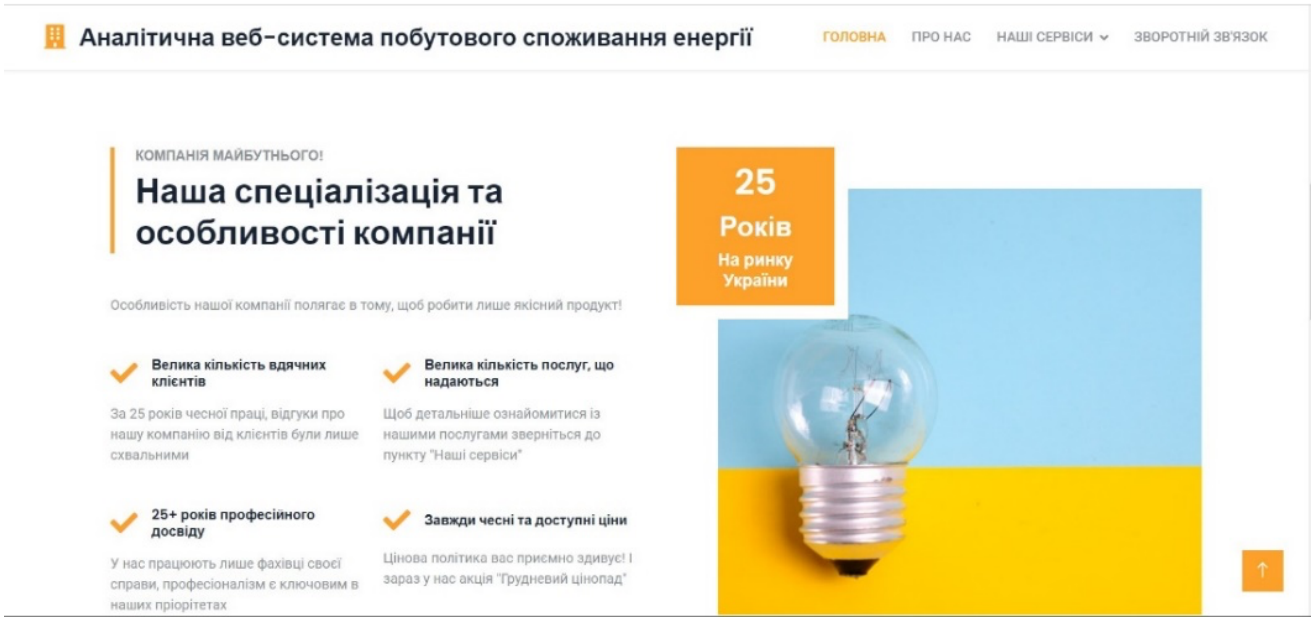


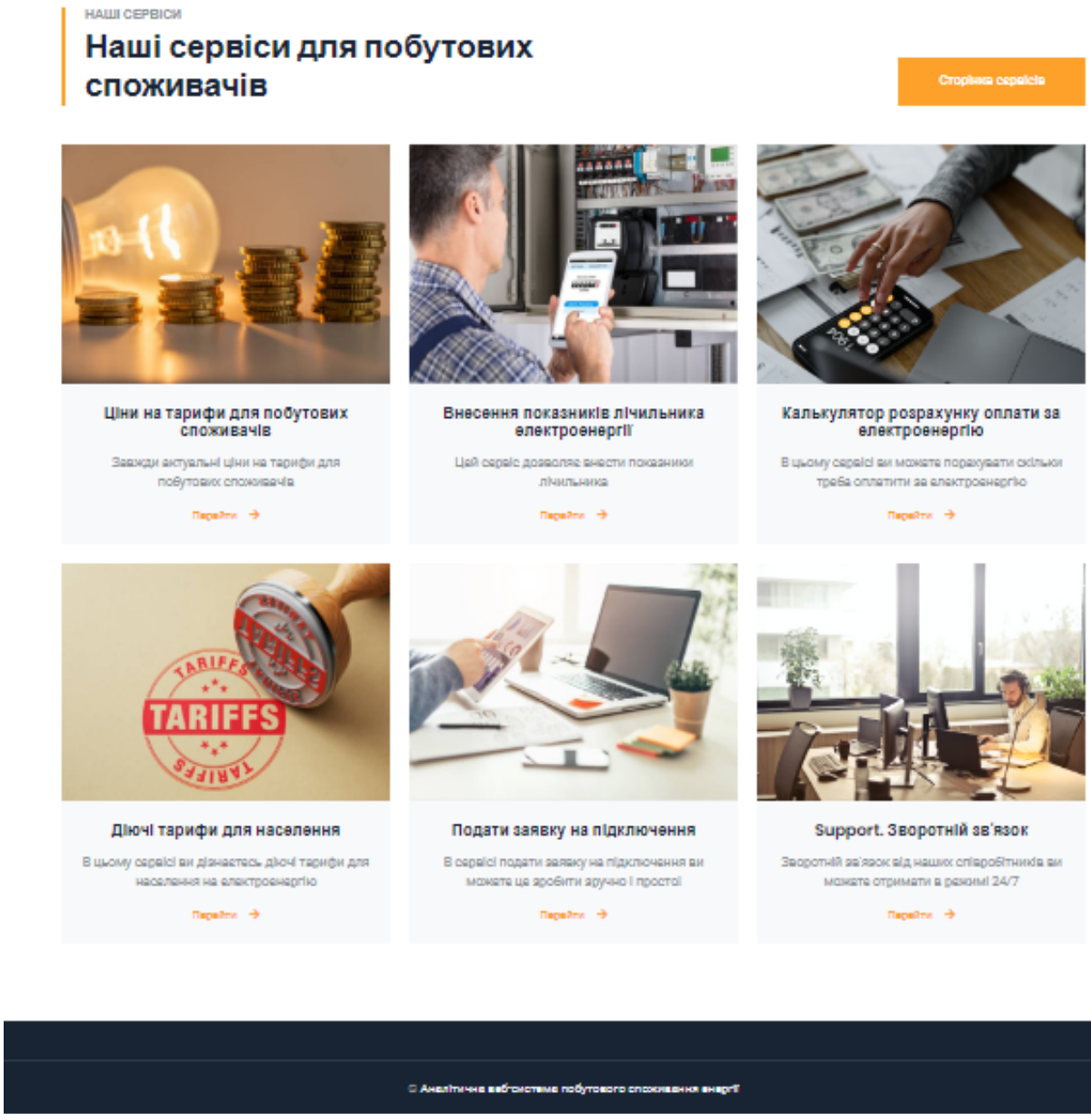
Схема взаємодії застосунку з базою даних



Головна сторінка



Сторінка «Наші сервіси»



Додаток В

(довідниковий)

Лістинг програми - верстка

```
<!DOCTYPE html>
<html lang="uk">
{% load static %}
<head>
    <meta charset="utf-8">
    <title>Аналітична веб-система побутового споживання
енергії</title>
    <meta content="width=device-width, initial-scale=1.0"
name="viewport">
    <meta content="" name="keywords">
    <meta content="" name="description">

    <!-- Favicon -->
    <link href="img/favicon.ico" rel="icon">

    <!-- Google Web Fonts -->
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500
&family=Poppins:wght@600;700&display=swap" rel="stylesheet">

    <!-- Icon Font Stylesheet -->
    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.10.0/css/all.min.css" rel="stylesheet">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.4.1/font/bootstrap-icons.css" rel="stylesheet">

    <!-- Libraries Stylesheet -->
    <link href="{% static 'lib/animate/animate.min.css' %}"
rel="stylesheet">
    <link href="{% static
'lib/owlcarousel/assets/owl.carousel.min.css' %}"
rel="stylesheet">

    <!-- Customized Bootstrap Stylesheet -->
    <link href="{% static 'css/bootstrap.min.css' %}"
rel="stylesheet">

    <!-- Template Stylesheet -->
    <link href="{% static 'css/style.css' %}" rel="stylesheet">
</head>

<body>
```

```
    <!-- Spinner Start -->
    <div id="spinner" class="show bg-white position-fixed
translate-middle w-100 vh-100 top-50 start-50 d-flex align-items-
center justify-content-center">
        <div class="spinner-grow text-primary"
role="status"></div>
    </div>
    <!-- Spinner End -->


 <!-- Navbar Start -->
    <nav class="navbar navbar-expand-lg bg-white navbar-light
sticky-top px-4 px-lg-5 py-lg-0">
        <a href="/" class="navbar-brand d-flex align-items-
center">
            <h3 class="m-0"><i class="fa fa-building text-primary
me-3"></i>Аналітична веб-система побутового споживання
енергії</h3>
        </a>
        <button type="button" class="navbar-toggler" data-bs-
toggle="collapse" data-bs-target="#navbarCollapse">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarCollapse">
            <div class="navbar-nav ms-auto py-3 py-lg-0">
                <a href="/" class="nav-item nav-link
active">Головна</a>
                <a href="/about" class="nav-item nav-link">Про
нас</a>
         <div class="nav-item dropdown">
            <a href="/service" class="nav-link dropdown-toggle"
data-bs-toggle="dropdown">Наші сервіси</a>
            <div class="dropdown-menu bg-light m-0">
               <a href="/price" class="dropdown-item">Ціни на
тарифи</a>
               <a href="/pokazi" class="dropdown-item">Внесення
показників</a>
               <a href="/calc" class="dropdown-item">Калькулятор
розрахунку</a>
               <a href="/tarif" class="dropdown-item">Діючі
тарифи</a>
               <a href="/appointment" class="dropdown-item">Заявка
на підключення</a>
            </div>
         </div>
         <a href="/support" class="nav-item nav-link">Зворотній
зв'язок</a>
            </div>
        </div>
    </nav>
    <!-- Navbar End -->

    <!-- Carousel Start -->
    <div class="container-fluid p-0 mb-5 wow fadeIn" data-wow-
```

```
delay="0.1s">
        <div id="header-carousel" class="carousel slide" data-bs-
ride="carousel">
            <div class="carousel-inner">
                <div class="carousel-item active">
                    <img class="w-100" src="{% static
'img/carousel-1.jpg' %}" alt="Image">
                    <div class="carousel-caption">
                        <div class="container">
                            <div class="row justify-content-
center">
                                <div class="col-12 col-lg-10">
                                    <h5 class="text-light text-
uppercase mb-3 animated slideInDown">Вітаємо в нашій</h5>
                                    <h1 class="display-2 text-
light mb-3 animated slideInDown">Аналітичній веб-системі
побутового споживання енергії</h1>
                                    <ol class="breadcrumb mb-4 pb-
2">
                                        <li class="breadcrumb-item
fs-5 text-light">Чесні</li>
                                        <li class="breadcrumb-item
fs-5 text-light">Відкриті</li>
                                        <li class="breadcrumb-item
fs-5 text-light">Доступні</li>
                                    </ol>
                                    <a href="/about" class="btn
btn-primary py-3 px-5">Дізнатися більше</a>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
                <div class="carousel-item">
                    <img class="w-100" src="{% static
'img/carousel-2.jpg' %}" alt="Image">
                    <div class="carousel-caption">
                        <div class="container">
                            <div class="row justify-content-
center">
                                <div class="col-12 col-lg-10">
                                    <h5 class="text-light text-
uppercase mb-3 animated slideInDown">Вас вітають</h5>
                                    <h1 class="display-2 text-
light mb-3 animated slideInDown">Найкращі фахівці в галузі
електроенергії</h1>
                                    <ol class="breadcrumb mb-4 pb-
2">
                                        <li class="breadcrumb-item
fs-5 text-light">Професійні</li>
                                        <li class="breadcrumb-item
fs-5 text-light">Надійні</li>
                                        <li class="breadcrumb-item
fs-5 text-light">Щирі</li>
```

```
                                    </ol>
                                    <a href="" class="btn btn-
primary py-3 px-5">Дізнатися більше</a>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
                <button class="carousel-control-prev" type="button"
data-bs-target="#header-carousel"
                    data-bs-slide="prev">
                    <span class="carousel-control-prev-icon" aria-
hidden="true"></span>
                    <span class="visually-hidden">Попередній</span>
                </button>
                <button class="carousel-control-next" type="button"
data-bs-target="#header-carousel"
                    data-bs-slide="next">
                    <span class="carousel-control-next-icon" aria-
hidden="true"></span>
                    <span class="visually-hidden">Наступний</span>
                </button>
            </div>
        </div>
        <!-- Carousel End -->


        <!-- Features Start -->
        <div class="container-xxl py-5">
            <div class="container">
                <div class="row g-5">
                    <div class="col-lg-6 wow fadeInUp" data-wow-
delay="0.1s">
                        <div class="border-start border-5 border-
primary ps-4 mb-5">
                            <h6 class="text-body text-uppercase mb-
2">Компанія майбутнього!</h6>
                            <h1 class="display-6 mb-0">Наша
спеціалізація та особливості компанії</h1>
                        </div>
                        <p class="mb-5">Особливість нашої компанії
полягає в тому, щоб робити лише якісний продукт! </p>
                        <div class="row gy-5 gx-4">
                            <div class="col-sm-6 wow fadeIn" data-wow-
delay="0.1s">
                                <div class="d-flex align-items-center
mb-3">
                                    <i class="fa fa-check fa-2x text-
primary flex-shrink-0 me-3"></i>
                                    <h6 class="mb-0">Велика кількість
вдячних клієнтів</h6>
                                </div>
                                <span>За 25 років чесної праці,
```

```
відгуки про нашу компанію від клієнтів були лише схвальними</span>
                        </div>
                        <div class="col-sm-6 wow fadeIn" data-wow-
delay="0.2s">
                            <div class="d-flex align-items-center
mb-3">
                                <i class="fa fa-check fa-2x text-
primary flex-shrink-0 me-3"></i>
                                <h6 class="mb-0">Велика кількість
послуг, що надаються</h6>
                            </div>
                            <span>Щоб детальніше ознайомитися із
нашими послугами зверніться до пункту "Наші сервіси"</span>
                        </div>
                        <div class="col-sm-6 wow fadeIn" data-wow-
delay="0.3s">
                            <div class="d-flex align-items-center
mb-3">
                                <i class="fa fa-check fa-2x text-
primary flex-shrink-0 me-3"></i>
                                <h6 class="mb-0">25+ років
професійного досвіду</h6>
                            </div>
                            <span>У нас працюють лише фахівці
своєї справи, професіоналізм є ключовим в наших пріорітетах</span>
                        </div>
                        <div class="col-sm-6 wow fadeIn" data-wow-
delay="0.4s">
                            <div class="d-flex align-items-center
mb-3">
                                <i class="fa fa-check fa-2x text-
primary flex-shrink-0 me-3"></i>
                                <h6 class="mb-0">Завжди чесні та
доступні ціни</h6>
                            </div>
                            <span>Цінова політика вас приємно
здивує! І зараз у нас акція "Грудневий цінопад"</span>
                        </div>
                    </div>
                </div>
                <div class="col-lg-6 wow fadeInUp" data-wow-
delay="0.5s">
                    <div class="position-relative overflow-hidden
ps-5 pt-5 h-100" style="min-height: 400px;">
                        <img class="position-absolute w-100 h-100"
src="{% static 'img/feature.jpg' %}" alt="" style="object-fit:
cover;">
                        <div class="position-absolute top-0 start-
0 bg-white pe-3 pb-3" style="width: 200px; height: 200px;">
                            <div class="d-flex flex-column
justify-content-center text-center bg-primary h-100 p-3">
                                <h1 class="text-white">25</h1>
                                <h2 class="text-white">Років</h2>
                                <h5 class="text-white mb-0">На
```

```
ринку України</h5>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    <!-- Features End -->


    <!-- Service Start -->
    <div class="container-xxl py-5">
        <div class="container">
            <div class="row g-5 align-items-end mb-5">
                <div class="col-lg-6 wow fadeInUp" data-wow-
delay="0.1s">
                    <div class="border-start border-5 border-
primary ps-4">
                        <h6 class="text-body text-uppercase mb-
2">Наші сервіси</h6>
                        <h1 class="display-6 mb-0">Наші сервіси
для побутових споживачів</h1>
                    </div>
                </div>
                <div class="col-lg-6 text-lg-end wow fadeInUp"
data-wow-delay="0.3s">
                    <a class="btn btn-primary py-3 px-5"
href="/services">Сторінка сервісів</a>
                </div>
            </div>
            <div class="row g-4 justify-content-center">
                <div class="col-lg-4 col-md-6 wow fadeInUp" data-
wow-delay="0.1s">
                    <div class="service-item bg-light overflow-
hidden h-100">
                        <img class="img-fluid" src="{% static
'img/service-1.jpg' %}" alt="">
                        <div class="service-text position-relative
text-center h-100 p-4">
                            <h5 class="mb-3">Ціни на тарифи для
побутових споживачів</h5>
                            <p>Завжди актуальні ціни на тарифи для
побутових споживачів</p>
                            <a class="small"
href="/price">Перейти<i class="fa fa-arrow-right ms-3"></i></a>
                        </div>
                    </div>
                </div>
                <div class="col-lg-4 col-md-6 wow fadeInUp" data-
wow-delay="0.3s">
                    <div class="service-item bg-light overflow-
hidden h-100">
                        <img class="img-fluid" src="{% static
```

```
'img/service-2.jpg' %}" alt="">
                        <div class="service-text position-relative
text-center h-100 p-4">
                            <h5 class="mb-3">Внесення показників
лічильника електроенергії</h5>
                            <p>Цей сервіс дозволяє внести
показники лічильника</p>
                            <a class="small"
href="/pokazi">Перейти<i class="fa fa-arrow-right ms-3"></i></a>
                        </div>
                    </div>
                </div>
                <div class="col-lg-4 col-md-6 wow fadeInUp" data-
wow-delay="0.5s">
                    <div class="service-item bg-light overflow-
hidden h-100">
                        <img class="img-fluid" src="{% static
'img/service-3.jpg' %}" alt="">
                        <div class="service-text position-relative
text-center h-100 p-4">
                            <h5 class="mb-3">Калькулятор
розрахунку оплати за електроенергію</h5>
                            <p>В цьому сервісі ви можете
порахувати скільки треба оплатити за електроенергію</p>
                            <a class="small"
href="/calc">Перейти<i class="fa fa-arrow-right ms-3"></i></a>
                        </div>
                    </div>
                </div>
                <div class="col-lg-4 col-md-6 wow fadeInUp" data-
wow-delay="0.1s">
                    <div class="service-item bg-light overflow-
hidden h-100">
                        <img class="img-fluid" src="{% static
'img/service-4.jpg' %}" alt="">
                        <div class="service-text position-relative
text-center h-100 p-4">
                            <h5 class="mb-3">Діючі тарифи для
населення</h5>
                            <p>В цьому сервісі ви дізнаєтесь діючі
тарифи для населення на електроенергію</p>
                            <a class="small"
href="/tarif">Перейти<i class="fa fa-arrow-right ms-3"></i></a>
                        </div>
                    </div>
                </div>
                <div class="col-lg-4 col-md-6 wow fadeInUp" data-
wow-delay="0.3s">
                    <div class="service-item bg-light overflow-
hidden h-100">
                        <img class="img-fluid" src="{% static
'img/service-5.jpg' %}" alt="">
                        <div class="service-text position-relative
text-center h-100 p-4">
```

```
                            <h5 class="mb-3">Подати заявку на
підключення</h5>
                            <p>В сервісі подати заявку на
підключення ви можете це зробити зручно і просто!</p>
                            <a class="small"
href="/appointment">Перейти<i class="fa fa-arrow-right ms-
3"></i></a>
                        </div>
                    </div>
                </div>
                <div class="col-lg-4 col-md-6 wow fadeInUp" data-
wow-delay="0.5s">
                    <div class="service-item bg-light overflow-
hidden h-100">
                        <img class="img-fluid" src="{% static
'img/service-6.jpg' %}" alt="">
                        <div class="service-text position-relative
text-center h-100 p-4">
                            <h5 class="mb-3">Support. Зворотній
зв'язок</h5>
                            <p>Зворотній зв'язок від наших
співробітників ви можете отримати в режимі 24/7</p>
                            <a class="small"
href="/support">Перейти<i class="fa fa-arrow-right ms-3"></i></a>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <!-- Service End -->


    <!-- Footer Start -->
    <div class="container-fluid bg-dark footer mt-5 pt-5 wow
fadeIn" data-wow-delay="0.1s">
        <div class="container-fluid copyright">
            <div class="container">
                <div class="row">
                    <div class=" text-center ">
                        &copy; <a href="#">Аналітична веб-система
побутового споживання енергії</a>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <!-- Footer End -->


    <!-- Back to Top -->
    <a href="#" class="btn btn-lg btn-primary btn-lg-square back-
to-top"><i class="bi bi-arrow-up"></i></a>
```

```
    <!-- JavaScript Libraries -->

    <script src="https://code.jquery.com/jquery-
3.4.1.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0/dist/js/bootstra
p.bundle.min.js"></script>
    <script src="{% static 'lib/wow/wow.min.js' %}"></script>
    <script src="{% static 'lib/easing/easing.min.js'
%}"></script>
    <script src="{% static 'lib/waypoints/waypoints.min.js'
%}"></script>
    <script src="{% static 'lib/owlcarousel/owl.carousel.min.js'
%}"></script>

    <!-- Template Javascript -->
    <script src="{% static 'js/main.js' %}"></script>
</body>

</html>
```

Лістинг основних методів та класів

admin.py

```
from django.contrib import admin
from bases.models import Price, Appointment, Pokazi, Support

admin.site.register(Price)

@admin.register(Appointment)
class AppointmentAdmin(admin.ModelAdmin):
    pass

@admin.register(Pokazi)
class PokaziAdmin(admin.ModelAdmin):
    pass

@admin.register(Support)
class SupportAdmin(admin.ModelAdmin):
    pass
```

models.py

```
from django.db import models

class Price(models.Model):
    name= models.TextField()
    one_class = models.CharField(max_length=30)
    two_class = models.CharField(max_length=30)

    @staticmethod
    def get_all_prices():
        return Price.objects.all()

    def __str__(self):
        return self.name

class Appointment(models.Model):
    first_name = models.CharField('Прізвище та ім\'я',
max_length=50)
    email = models.CharField('email', max_length=50)
    adresa = models.CharField('Ваша адреса', max_length=50)
    more_information = models.TextField('Вкажіть бажаний час або
ще якусь додаткова інформація')
    data_public = models.DateField('Дата подачі заявки')

    def __str__(self):
        return self.first_name

class Pokazi(models.Model):
    pokazi_lichilnyka = models.CharField('Покази лічильника',
```

```
max_length=50)
    data_public = models.DateField('Дата подачі показів')

    def __str__(self):
        return self.pokazi_lichilnyka

class Support(models.Model):
    first_name = models.CharField('Прізвище та ім\'я',
max_length=50)
    email = models.CharField('email', max_length=50)
    tema = models.CharField('Тема', max_length=50)
    message = models.TextField('Повідомлення')

    def __str__(self):
        return self.first_name
```

forms.py

```
from .models import Appointment, Pokazi, Support
from django.forms import ModelForm, TextInput, Textarea, DateInput


class SupportForm(ModelForm):
    class Meta:
        model = Support
        fields = ['first_name', 'email', 'tema', 'message']

        widgets = {
            'first_name': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Прізвище та ім\'я',
                'id': "gname",
            }),
            'email': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': "Email",
                'id': "gmail",
            }),
            'tema': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Тема',
                'id': "tema",
            }),
            'message': Textarea(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Повідомлення',
                'id': "message",
            }),
```

```
        }

class AppointmentForm(ModelForm):
    class Meta:
        model = Appointment
        fields = ['first_name', 'email', 'adresa',
'more_information', 'data_public']

        widgets = {
            'first_name': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Прізвище та ім\'я',
                'id': "gname",
            }),
            'email': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': "Email",
                'id': "gmail",
            }),
            'adresa': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Адреса підключення',
                'id': "adresa",
            }),
            'more_information': Textarea(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Додаткові відомості',
                'id': "message",
            }),
            'data_public': DateInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Бажана дата підключення у форматі
01.01.2023',
                'id': "date_appointment",
            }),
        }

class PokaziForm(ModelForm):
    class Meta:
        model = Pokazi
        fields = ['pokazi_lichilnyka', 'data_public']

        widgets = {
            'pokazi_lichilnyka': TextInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Покази лічильника',
                'id': "lichilnyk_pokazi",
            }),
```

```
            'data_public': DateInput(attrs={
                'class': 'form-control border-0 col-sm-3 form-
floating',
                'placeholder': 'Покази на дату у форматі
01.01.2023',
                'id': "date_pokazi",
            }),

        }
```

views.py

```
from django.shortcuts import render
from bases.models import Price
from .forms import AppointmentForm, PokaziForm, SupportForm

def index(request):
    return render(request, 'bases/index.html')

def about(request):
    return render(request, 'bases/about.html')

def service(request):
    return render(request, 'bases/service.html')

def price(request):
    price_data = Price.objects.all()
    return render(request, 'bases/price.html', {'price_data' :
price_data})

def pokazi(requests):
    error = ''
    if requests.method == 'POST':
        form = PokaziForm(requests.POST)
        if form.is_valid():
            form.save()
        else:
            error = 'Форма заповнена неправильно'
    form = PokaziForm()
    data = {
        'form': form,
        'error': error,
    }
    return render(requests, 'bases/pokazi.html', data)

def calc(request):
    return render(request, 'bases/calc.html')

def tarif(request):
    return render(request, 'bases/tarif.html')

def appointment(requests):
    error = ''
    if requests.method == 'POST':
```

```
            form = AppointmentForm(requests.POST)
            if form.is_valid():
                form.save()
            else:
                error = 'Форма заповнена неправильно'
        form = AppointmentForm()
        data = {
            'form': form,
            'error': error,
        }
        return render(requests, 'bases/appointment.html', data)

    def support(requests):
        error = ''
        if requests.method == 'POST':
            form = SupportForm(requests.POST)
            if form.is_valid():
                form.save()
            else:
                error = 'Форма заповнена неправильно'
        form = SupportForm()
        data = {
            'form': form,
            'error': error,
        }
        return render(requests, 'bases/support.html', data)
```

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: «Аналітична веб-система побутового споживання енергії»

Тип роботи: Магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ КСУ, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 96,3% Схожість 3,7%

Аналіз звіту подібності (відмітити потрібне)

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

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

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

Особа, відповідальна за перевірку _____ Галущак А.В.
(підпис) (прізвище, ініціали)

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

Автор роботи _____ Саранчук С.В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Дубовой В.М.
(підпис) (прізвище, ініціали)