

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра інформаційних управляючих систем та технологій

«Допущений до захисту»

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

«\_\_\_» \_\_\_\_\_ 2025 р.

*КВАЛІФІКАЦІЙНА РОБОТА*

на здобуття ступеня вищої освіти «бакалавр»

на тему: «Розробка інформаційної підсистеми кабінету користувача інтернет  
провайдера»

Виконав: студент 4148ст3 групи



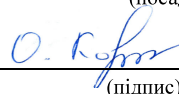
(підпис)

*Рустамов М.М.*

Керівник роботи:

Викладач к.т.н \_\_\_\_\_

(посада, науковий ступень вчене звання)



(підпис)

*Гайдаєнко О.В.*

Миколаїв – 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра інформаційних управляючих систем та технологій

Спеціальність 124 "Системний аналіз"

Освітня програма "Системний аналіз"

«ЗАТВЕРДЖУЮ»

Гарант освітньої програми



к.т.н, доц. Гайдаєнко О.В.

«\_\_\_» \_\_\_\_\_ 2025 р.

*ЗАВДАННЯ*

*НА КВАЛІФІКАЦІЙНУ РОБОТУ*

на здобуття ступеня вищої освіти «бакалавр»

Студенту Рустомову Миколі Михайловичу

(Прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка інформаційної підсистеми кабінету користувача  
інтернет провайдера»

Керівник роботи К.Т.Н, доцент. Гайдаєнко О.В.

Затверджені наказом ректора № 295-уч від «27» березня 2025 року.

2. Термін подання роботи: 20 червня 2025 р.

3. Вихідні дані до проекту (роботи): ДСТУ щодо обробки інформації, літературні  
джерела, технічна документація на існуючі аналоги систем,

4. Перелік питань, що належать до розробки (найменування розділів): Титульний  
аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та 3  
англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та  
термінів (при необхідності); Вступ; Опис предметної галузі та постановка задачі;  
Проект програмного забезпечення (ескізний, технічний та робочий); Результати  
розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список  
використаних джерел; Додатки (технічне завдання, текст програми, опис  
програми, інструкція користувача, програма та методика випробувань  
програмного забезпечення).

5. Перелік презентаційних матеріалів 1. Мета роботи. 2. Завдання роботи. 3. Об'єкт дослідження. 4. Схема організаційної структури підприємства. 5. Діаграма варіантів використання. 6. Концептуальна та фізичні моделі даних. 7. Програмні та технічні засоби. 8. Екранні форми.

#### 6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата   |                  |
|--------|-------------------------------------------|----------------|------------------|
|        |                                           | завдання видав | завдання прийняв |
| 1      | Гайдаєнко О.В.                            | 03.02.2025     | 22.02.2025       |
| 2      | Гайдаєнко О.В.                            | 22.02.2025     | 21.03.2025       |
| 3      | Гайдаєнко О.В.                            | 21.03.2025     | 25.04.2025       |
| 4      | Гайдаєнко О.В.                            | 25.04.2025     | 07.06.2025       |

7. Дата видачі завдання 03 лютого 2025 р.

#### КАЛЕНДАРНИЙ ПЛАН

| Номер | Назва етапів роботи                    | Терміни виконання етапів роботи | Примітка |
|-------|----------------------------------------|---------------------------------|----------|
| 1.    | Аналіз предметної галузі               | 13.02.2025                      |          |
| 2.    | Постановка задачі                      | 15.02.2025                      |          |
| 3.    | Розробка концепції                     | 22.02.2025                      |          |
| 4.    | Розробка проєкту ІС                    | 21.03.2025                      |          |
| 5.    | Реалізація проєкту                     | 25.04.2025                      |          |
| 6.    | Розробка питань з охорони праці        | 07.06.2025                      |          |
| 7.    | Розробка графічних матеріалів          | 12.06.2025                      |          |
| 8.    | Подання роботи на перевірку на плагіат | 10.06.2025                      |          |
| 9.    | Подання роботи рецензенту              | 15.06.2025                      |          |
| 10.   | Подання роботи на захист               | 20.06.2025                      |          |

Студент

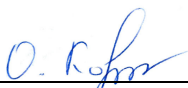


(підпис)

Рустамов М.М.

(ПІБ)

Керівник роботи



(підпис)

Гайдаєнко О.В.

(ПІБ)

## АНОТАЦІЯ

Кваліфікаційна робота бакалавра «Розробка інформаційної підсистеми кабінету користувача інтернет провайдера»: с. 67, рис. 23, 16 джерел.

ІНТЕРНЕТ-ПРОВАЙДЕР, JAVA, CSS, HTML, SQL.

Мета роботи: розробка вебсайту інтернет-провайдера “SkyNet”.

Метод дослідження – аналітичний.

Для розробки сайту та написання дипломної роботи були вивчені та розглянуті засоби для створення Web-сайтів.

Під час виконання завдання були розглянуті основи створення сайтів, вивчена теорія, що пізніше була реалізована на практиці у вигляді створеного програмного продукту. Для створення сайту було використано HTML, CSS, JavaScript.

Результатом даної роботи є веб-сайт інтернет-провайдера “SkyNet”.

**Ключові слова:** вебсайт, інтернет-провайдер, розробка сайту, HTML, CSS, JavaScript, SQL, користувацький інтерфейс, адаптивний дизайн, SkyNet.



## SUMMARY

Bachelor's qualifying paper «Development of the Internet Provider's Website»: 67 pages, 23 figures, 16 references.

INTERNET-PROVIDER, JAVA, CSS, HTML, SQL.

The object of the study is the main means of developing modern websites.

The aim of the study is the website of the Internet provider «SkyNet».

The method of research is analytical.

For the development of the website and the writing of the thesis, the tools for creating websites were studied and considered.

During the execution of the task, the basics of website creation were considered, the theory was studied, which was later implemented in practice and in the form of a created software product. HTML, CSS, JavaScript were used to create the site.

The result of this work is the development of the website of the Internet provider «SkyNet».

**Keywords:** website, Internet provider, website development, HTML, CSS, JavaScript, SQL, custom interface, adaptive design, SkyNet.

## ЗМІСТ

|                                                                           |    |
|---------------------------------------------------------------------------|----|
| ВСТУП .....                                                               | 7  |
| РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ .....            | 8  |
| 1.1 Аналіз предметної області.....                                        | 8  |
| 1.2 Огляд існуючих систем і їх недоліків.....                             | 10 |
| 1.3 Концепція системи .....                                               | 14 |
| 1.4 Формування вимог до сайту .....                                       | 16 |
| 1.5 Постановка задачі.....                                                | 17 |
| РОЗДІЛ 2. Проектування інформаційної підсистеми кабінету користувача..... | 22 |
| 2.1 Загальносистемні рішення.....                                         | 22 |
| 2.2 Концептуальна модель.....                                             | 23 |
| 2.3 Діаграма послідовності .....                                          | 26 |
| 2.4 Логічна модель .....                                                  | 28 |
| 2.5 Фізична модель.....                                                   | 30 |
| РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ.....                          | 34 |
| 3.1 Діаграма розгортання системи.....                                     | 34 |
| 3.2 Рішення щодо апаратного забезпечення .....                            | 36 |
| 3.3 Рішення щодо інформаційного забезпечення.....                         | 37 |
| 3.4 Екранні форми розробки .....                                          | 41 |
| Розділ 4. Охорона праці .....                                             | 46 |
| 4.1 Шкідливі та небезпечні фактори праці програміста .....                | 46 |
| 4.2 Ергономічні та санітарні вимоги до робочого місця .....               | 47 |
| 4.3 Нормативні вимоги до організації праці .....                          | 48 |
| 4.4 Щоденні процедури та заборони .....                                   | 48 |
| 4.5 Покарання за невідповідність .....                                    | 49 |
| ВИСНОВКИ.....                                                             | 50 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....                                          | 51 |
| ДОДАТОК А - ІНСТРУКЦІЯ КОРИСТУВАЧА .....                                  | 51 |
| ДОДАТОК Б - ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ .....                           | 58 |
| ДОДАТОК В - ФРАГМЕНТ ПРОГРАМНОГО КОДУ .....                               | 60 |

## ВСТУП

Створення вебсайтів зараз дуже популярне, і з кожним роком цей процес стає простішим. Тепер будь-хто може зробити власний сайт за кілька годин.

Сьогодні існує багато онлайн-конструкторів, таких як Wix, Google AdSense, Flexbe, uKit та інші. Вони дозволяють швидко створювати сайти, але мають обмежений функціонал, оскільки базуються на готових шаблонах. Через це великі й складні проекти потребують використання професійних технологій: мови гіпертекстової розмітки HTML, стилів CSS, програмування на JavaScript, роботи з базами даних через SQL та інших інструментів.

З кожним роком мови програмування стають доступнішими для вивчення, а в інтернеті з'являється безліч ресурсів для навчання. Завдяки цьому майже кожен може опанувати необхідні знання та створити власний сайт.

Мета дослідження – розробка кабінету користувача інтернет-провайдера “SkyNet”.

Для реалізації поставленої мети було сформульовано наступні завдання:

- проаналізувати вебсайти інтернет провайдерів;
- дослідити та обрати засоби розробки;
- розробити проєкт сайту інтернет-провайдера “SkyNet”;
- розробити сайт відповідно до проєкту.

Розроблена система задовольняє всім вимогам, що були зазначені на етапі постановки завдання.

Кваліфікаційна робота складається зі вступу, чотирьох розділів.

Перший розділ містить основні відомості про особистий кабінет інтернет-провайдера, а саме розглянуто сайти-аналоги та сформульовано основні вимоги до розроблюваного сайту, досліджено інструменти розробки.

Другий розділ містить технічне завдання, опис структури вебсайтів, діаграму варіантів використання, макети для створення сайту.

Третій розділ містить опис розробленого вебсайту, особливості реалізації, опис вебсторінок розробленої інформаційної системи.

## **РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ**

### **1.1 Аналіз предметної області**

Інтернет-провайдер «SkyNet» здійснює надання телекомунікаційних послуг, зокрема доступу до мережі Інтернет, для фізичних осіб та організацій. На момент проведення дослідження підприємство не має сучасного вебресурсу, що ускладнює процес інформування клієнтів, просування послуг та зворотний зв'язок із користувачами. У зв'язку з цим виникає об'єктивна потреба у створенні багатофункціонального, зручного та адаптивного вебсайту.

Основною метою створення зазначеного ресурсу є реалізація платформи, яка дозволить компанії ефективно презентувати свої послуги в онлайн-середовищі, забезпечити зручний доступ до актуальної інформації щодо тарифів, умов підключення, а також організувати механізми інтерактивної взаємодії з користувачами.

Запровадження інформаційної підсистеми у вигляді особистого кабінету забезпечить централізований канал комунікації, через який клієнти зможуть залишати заявки, контролювати стан рахунку, звертатися до технічної підтримки тощо. Такий інструмент дозволить суттєво покращити якість обслуговування, підвищити рівень довіри до компанії та розширити клієнтську базу.

Зважаючи на стрімке зростання частки мобільних користувачів, сайт має бути адаптованим до різних типів пристроїв. Це дозволить забезпечити належну доступність сервісів для широкого кола споживачів. Інтерфейс повинен бути інтуїтивно зрозумілим, доступним для користувачів з різним рівнем комп'ютерної підготовки, а також містити інтерактивні елементи, такі як форма зворотного зв'язку, онлайн-чат, авторизація тощо.

Для адміністрування контенту передбачається впровадження системи керування вмістом (CMS), що дасть змогу оперативно оновлювати інформацію про послуги, акції, тарифи, а також забезпечити масштабованість у майбутньому.

Аналіз функціональності існуючих особистих кабінетів інших інтернет-провайдерів свідчить про наявність низки характерних недоліків, серед яких:

1. Обмежений функціонал: Більшість особистих кабінетів дозволяють лише переглядати баланс або оплачувати послуги, але не надають можливості керувати тарифами, переглядати детальну історію платежів, налаштовувати обладнання чи замовляти додаткові послуги (наприклад, статичну IP-адресу). Це змушує користувачів звертатися до техпідтримки для виконання простих операцій, що знижує ефективність системи.

2. Незручний інтерфейс: Інтерфейси багатьох кабінетів є застарілими, перевантаженими інформацією або погано структурованими. Наприклад, важливі функції (зміна тарифу чи перевірка статусу заявки) часто заховані в підменю, що ускладнює їх пошук, особливо для недосвідчених користувачів.

3. Відсутність адаптивності: Багато особистих кабінетів погано оптимізовані для мобільних пристроїв, що створює незручності для користувачів, які заходять із смартфонів. Наприклад, елементи управління можуть бути занадто дрібними або неправильно відображатися на маленьких екранах.

4. Слабка інтерактивність: У більшості кабінетів відсутні інтерактивні елементи, такі як онлайн-чати чи інтеграція з месенджерами (Telegram, Viber), що змушує користувачів телефонувати до техпідтримки або чекати відповіді через електронну пошту.

5. Недостатня безпека: Деякі кабінети не забезпечують належного захисту даних користувачів, наприклад, через відсутність двофакторної автентифікації чи слабе шифрування при передачі платіжної інформації, що знижує довіру до системи.

6. Відсутність персоналізації: Кабінети рідко пропонують персоналізовані рекомендації, наприклад, щодо вибору оптимального тарифу на основі історії використання, що могло б покращити користувацький досвід.

Ці недоліки призводять до низької ефективності взаємодії клієнтів із провайдером, зниження їхньої задоволеності та ускладнення управління

послугами. Для подолання цих проблем особистий кабінет SkyNet має бути розроблений із урахуванням сучасних стандартів UX/UI, забезпечувати широкий функціонал, адаптивність, безпечну обробку даних і інтеграцію з інтерактивними каналами зв'язку, що зробить його зручним і конкурентоспроможним рішенням.

## 1.2 Огляд існуючих систем і їх недоліків

Тема кваліфікаційної роботи - розробка інформаційної підсистеми кабінету користувача інтернет-провайдерів - зумовлена проблемами, виявленими при аналізі сайтів провайдерів Kyivstar, Tenet та Дикий Сад, зокрема щодо функціональності їхніх особистих кабінетів. Основна проблематика полягає у обмеженій функціональності, незручних інтерфейсах, слабкій адаптивності та недостатній інтерактивності кабінетів, що знижує ефективність взаємодії клієнтів із провайдером. Нижче наведено аналіз особистих кабінетів цих провайдерів на основі доступної інформації з вебресурсів.

Особистий кабінет Kyivstar представлено на рисунку 1.1.

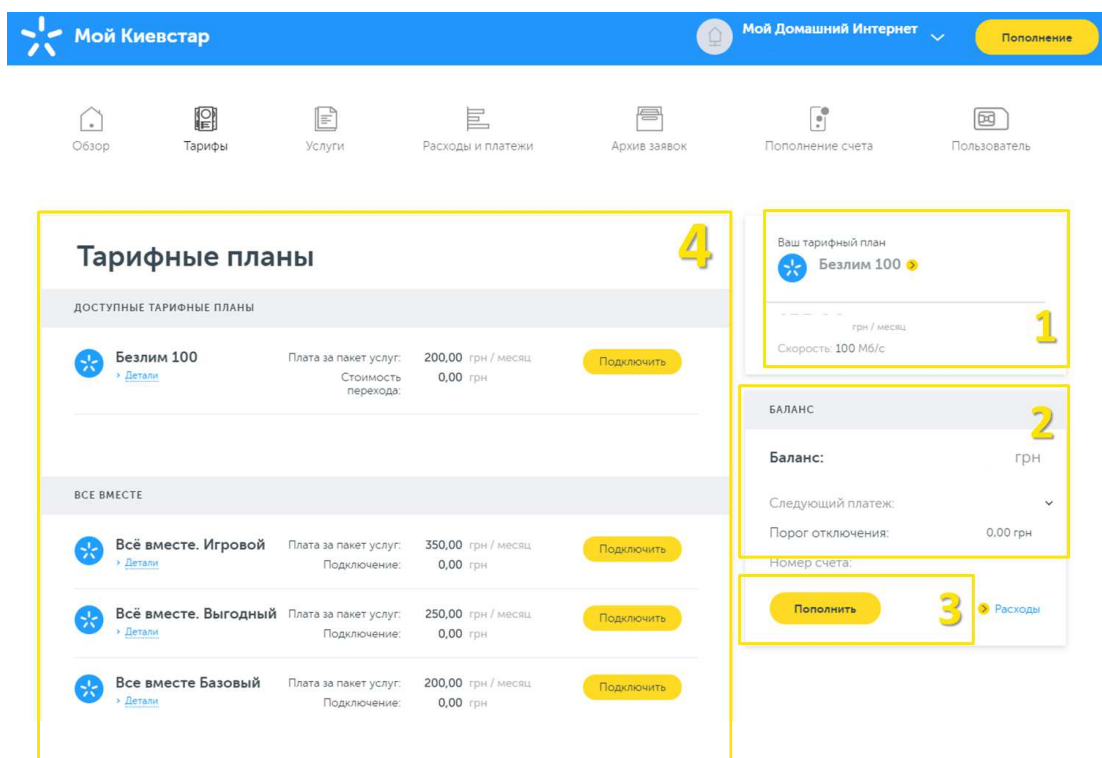


Рисунок 1.1 – Особистий кабінет

Розглянемо переваги та недоліки особистого кабінету Kyivstar

Переваги:

- Зручний доступ через додаток із постійним паролем;
- базові функції управління тарифами та платежами;
- інтеграція з програмою лояльності;
- підтримка eSIM для зручного переносу номера.

Недоліки:

- Обмежений функціонал (наприклад, відсутність налаштувань обладнання чи детальної історії платежів);
- одноразовий пароль для входу через сайт ускладнює регулярне використання;
- слабка інтеграція з месенджерами для підтримки.

Тепер розглянемо особистий кабінет інтернет провайдера «Tenet».

Особистий кабінет Tenet доступний на сайті [tenet.ua](http://tenet.ua) та через мобільний додаток “my TENET”. Користувачі можуть увійти, використовуючи номер особового рахунку та пароль.

Кабінет дозволяє перевірити стан рахунку, поповнити баланс, замовити послуги (інтернет, ТБ, мобільний зв’язок), переглянути історію платежів і налаштувати автоматичні платежі. Також підтримується Email- та SMS-інформування про стан рахунку (безкоштовно).

Кабінет інтегрований із реферальною програмою, яка пропонує бонуси (до 150 грн) за залучення друзів.

Технічна підтримка доступна через месенджери (Viber, Telegram) і телефон, але прямий чат у кабінеті відсутній.

Особистий кабінет представлений на рисунку 1.2.

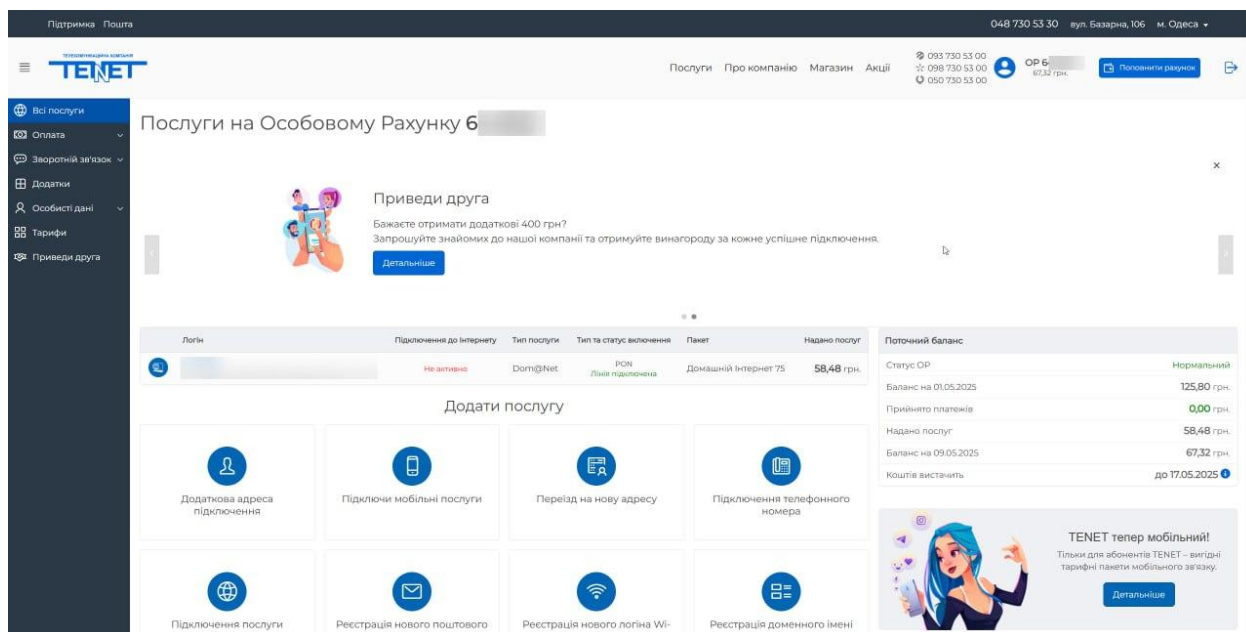


Рисунок 1.2 – Шапка сайту Tenet

Розглянемо переваги та недоліки особистого кабінету Tenet

Переваги:

- Широкий функціонал;
- Гнучкість для фізичних осіб;
- Спеціальні послуги для домашніх пакетів;
- Підтримка юридичних осіб;
- Деталізація статистики;

Недоліки:

- Складність інтерфейсу;
- Потенційна затримка в обробці заявок
- Перевантаження інформацією.



Розглянемо особистий кабінет інтернет провайдера «Дикий Сад».

Особистий кабінет Дикий Сад доступний через сайт wildpark.net для оплати послуг. Для входу потрібен номер особового рахунку та пароль. Кабінет дозволяє переглядати баланс, поповнювати рахунок, перевіряти статус підключення та подавати заявки на додаткові послуги (наприклад, інтерактивне чи цифрове ТБ). Проте кабінет на офіційному сайті має обмежений функціонал і не підтримує розширені функції, такі як зміна тарифів чи налаштування обладнання. Техпідтримка доступна через телефон, Telegram або форум провайдера, але прямий чат у кабінеті відсутній.

Особистий кабінет представлений на рисунку 1.3.

The screenshot displays the 'Дикий Сад' (Wild Park) personal account interface. At the top, there is a navigation bar with the provider's logo, a login field labeled 'Логін:', and a password field labeled 'О/Р:'. The main content area is divided into several sections:

- Стан рахунку** (Account Status): Shows the current account balance as 'Поточний стан рахунку, грн' with a value of **1358.30**.
- Поповнення рахунку** (Account Refill): Includes a form for entering the 'Сума' (Amount) and selecting a 'Платіжна система' (Payment System), with a 'Сплата' (Payment) button.
- Тариф та сервіси** (Tariff and Services): A table showing service details:

| Тарифний план | Домашній Інтернет  |
|---------------|--------------------|
| Вартість      | <b>260 грн/міс</b> |
| Швидкість     | 100/100 Mbps       |
| Закінчення    | 19.03.2024 10:53   |
| До закінчення | 18дн 18г 25хв      |
| Фіксована IP  | 100 грн/міс        |
- Анкетні дані** (Survey Data): A table showing user profile information:

|                  |                            |
|------------------|----------------------------|
| Назва            | Петренко Степан Андрійович |
| Особовий рахунок | [Redacted]                 |
| Адреса           | пр. Центральний, 52, кв. 4 |
| Тел.             | 067-1234567                |
| Email            | [Redacted]@mksat.net       |
| Реєстрація       | 07.06.2013                 |

Рисунок 1.3 – Особистий кабінет

Розглянемо переваги та недоліки особистого кабінету Дикого Саду

Переваги:

- підтримка шаблонів і автоплатежів;
- доступ до базових функцій перегляду балансу та подачі заявок.

Недоліки:

- Застарілий інтерфейс кабінету на сайті,
- обмежений функціонал (немає зміни тарифів чи історії платежів),
- відсутність адаптивності для мобільних пристроїв, слабка інтеграція з месенджерами чи чатом у кабінеті.

### 1.3 Концепція системи

На основі аналізу існуючих вебсайтів інтернет-провайдерів, зокрема Kyivstar, Tenet та Дикий сад, було встановлено, що жоден із них повністю не задовольняє сучасні вимоги до зручності користування, адаптивності, актуальності контенту та інтерактивності.

Основні недоліки існуючих сайтів:

- Незручна навігація.
- Відсутність або слабка мобільна версія.
- Складно знайти актуальну інформацію про тарифи та послуги.
- Відсутність інтерактивних елементів (онлайн-чати, форми зворотного зв'язку).
- Відсутність карти покриття або складна система перевірки доступності підключення.
- Застарілий або перевантажений дизайн.

У зв'язку з цим виникла потреба у створенні власного рішення - вебсайту інтернет-провайдера SkyNet, який усуває виявлені недоліки і враховує сучасні вимоги користувачів.

Основні особливості концепції вебсайту SkyNet:

- Адаптивний сучасний дизайн, що однаково добре виглядає на комп'ютерах і мобільних пристроях.

- Зрозуміла структура навігації.
- Актуальна інформація про послуги і тарифи.
- Інтерактивні елементи: форми зворотного зв'язку, онлайн-чат, швидка подача заявки на підключення.

- Карта покриття із зручним пошуком за адресою.
- Безпечна передача персональних даних.
- Використання CMS для легкого оновлення контенту без залучення розробників.

Таким чином, вебсайт **SkyNet** буде повністю відповідати сучасним стандартам і потребам цільової аудиторії, що вигідно відрізнятиме його від конкурентів.

Таблиця 1.1- відповідність вимогам

| Критерій                                        | Kyivstar | Tenet | Дикий сад | SkyNet |
|-------------------------------------------------|----------|-------|-----------|--------|
| Адаптивність сайту                              | ✓        | ✓     | ✗         | ✓      |
| Сучасний інтерфейс                              | ✗        | ✓     | ✗         | ✓      |
| Інтерактивність (чат, форма зворотного зв'язку) | ✗        | ✗     | ✗         | ✓      |
| Наявність карти покриття                        | ✓        | ✗     | ✗         | ✓      |
| Зручність пошуку інформації                     | ✗        | ✓     | ✗         | ✓      |
| Актуальність та легкість оновлення контенту     | ✗        | ✗     | ✗         | ✓      |
| Безпека даних користувачів                      | ✓        | ✓     | ✓         | ✓      |

#### 1.4 Формування вимог до сайту

На основі аналізу вебресурсів провідних інтернет-провайдерів, таких як Tenet, Kyivstar і «Дикий Сад», а також з урахуванням актуальних потреб та очікувань сучасної аудиторії користувачів, визначено основні функціональні та структурні вимоги до вебсайту компанії SkyNet.

Однією з ключових вимог є забезпечення інтуїтивно зрозумілої навігації, яка дає змогу оперативно знаходити необхідну інформацію. Центральним елементом інтерфейсу повинне стати верхнє меню, розміщене в шапці вебсторінки, яке передбачає логічне групування основних розділів: «Про нас», «Тарифи», «Підключення», «Підтримка», «Контакти» тощо. Така структура сприятиме зручній орієнтації на сайті для користувачів з різним рівнем цифрової грамотності.

З метою ефективної презентації актуальної інформації передбачається впровадження інтерактивного слайдера на головній сторінці. Цей елемент дозволяє акцентувати увагу відвідувачів на діючих акціях, оновлених тарифних планах або змінах у сервісному обслуговуванні. Динамічний візуальний контент не лише підвищує зацікавленість користувачів, але й формує враження постійного оновлення вебресурсу.

Важливою функціональною складовою сайту є інтеграція кнопки швидкого переходу до зовнішнього ресурсу — персонального кабінету користувача. Це забезпечить оперативний доступ до керування підключеними послугами, перегляду фінансових операцій і замовлення додаткових сервісів без необхідності додаткового пошуку відповідних сторінок.

Сайт має реалізовувати принципи адаптивного дизайну, що гарантує його коректне відображення на різноманітних пристроях — стаціонарних комп'ютерах, планшетах і смартфонах. Такий підхід сприяє забезпеченню зручного доступу до ресурсу незалежно від розмірів екрана або типу браузера. При цьому дизайн повинен відповідати сучасним стандартам UX/UI, що підвищить загальну якість користувацького досвіду.

Окрему увагу необхідно приділити доступності та структурованості контенту. Інформаційне наповнення має бути логічно організованим, лаконічним і поданим зрозумілою мовою. Доцільним є впровадження функціоналу пошуку за ключовими словами для пришвидшення знаходження потрібної інформації.

У підсумку, вебсайт компанії SkyNet повинен відповідати сучасним стандартам веброзробки, поєднуючи естетичну привабливість, функціональність, високу швидкодію та інтерактивність. Такий ресурс стане не лише цифровим представництвом компанії у мережі, а й основним інструментом комунікації з клієнтами, що забезпечить її конкурентоспроможність на ринку телекомунікаційних послуг.

### 1.5 Постановка задачі

На основі проведеного аналізу предметної області встановлено, що наявні вебресурси інтернет-провайдерів, зокрема їхні інформаційні підсистеми особистих кабінетів користувачів, характеризуються низкою критичних недоліків. Серед основних проблем слід виокремити обмеженість функціоналу (відсутність історії фінансових транзакцій, можливостей налаштування обладнання, інтеграції з месенджерами), неефективну навігацію, застарілий інтерфейс, недостатню адаптивність до мобільних платформ, обмежену інтерактивність (наприклад, відсутність онлайн-чатів або форм зворотного зв'язку), а також неналежний рівень безпеки обробки персональних даних. У сукупності ці чинники негативно впливають на якість взаємодії користувачів із системою, ускладнюють управління послугами й знижують конкурентоспроможність компаній на телекомунікаційному ринку.

З метою подолання вказаних недоліків передбачається розроблення сучасної інформаційної підсистеми особистого кабінету користувача для компанії SkyNet, яка відповідатиме актуальним вимогам галузі веброзробки, забезпечуватиме безпечний доступ до керування послугами та

вирізнятиметься зручним, адаптивним і естетично привабливим інтерфейсом. Проектна діяльність охоплює реалізацію наступних етапів:

1. Аналіз вимог та постановка задачі:
  - Детальне вивчення потреб цільової аудиторії (користувачів та адміністраторів).
  - Формалізація функціональних і нефункціональних вимог до особистого кабінету (наприклад, управління тарифами, перегляд історії платежів, інтеграція з месенджерами).
  - Визначення технічних обмежень і стандартів безпеки.
2. Дослідження та вибір технологій:
  - Аналіз інструментів розробки (Java, Spring Boot, MariaDB, Bootstrap тощо).
  - Вибір фреймворків, бібліотек і плагінів для забезпечення ефективної реалізації (Spring Security для автентифікації, Thymeleaf для шаблонів).
  - Оцінка сумісності технологій із вимогами адаптивності та безпеки.
3. Проектування архітектури системи:
  - Розробка структури бази даних для зберігання інформації про користувачів, тарифи, платежі тощо.
  - Створення діаграм (UML, ERD) для моделювання взаємодії компонентів системи.
  - Визначення API для інтеграції кабінету з зовнішніми сервісами (наприклад, платіжними системами).
4. Розробка прототипу інтерфейсу:
  - Створення макетів (wireframes) та дизайн-прототипів інтерфейсу кабінету з урахуванням принципів UX/UI.
  - Проектування адаптивного дизайну для коректного відображення на різних пристроях.

- Узгодження прототипу з вимогами замовника.
- 5. Розробка програмного забезпечення:
  - Налаштування середовища розробки (IntelliJ IDEA, Maven, Git).
  - Реалізація бекенду (Spring Boot, Spring Data JPA, Hibernate) для обробки запитів і управління даними.
  - Розробка фронтенду для створення інтерактивного інтерфейсу.
  - Інтеграція Spring Security для автентифікації та авторизації користувачів.
- 6. Тестування системи:
  - Проведення модульного тестування компонентів (наприклад, тестування API за допомогою Postman).
  - Інтеграційне тестування для перевірки взаємодії між бекендом і фронтендом.
  - Тестування адаптивності, безпеки (захист від SQL-ін'єкцій, XSS) та продуктивності.
  - Усунення виявлених помилок.
- 7. Впровадження та розгортання:
  - Налаштування серверного середовища (наприклад, Tomcat через XAMPP).
  - Розгортання бази даних (MariaDB) та забезпечення її резервного копіювання.
  - Запуск сайту на хостингу та перевірка коректності роботи в реальних умовах.
- 8. Документування та навчання:
  - Складання технічної документації (опис API, структури бази даних, інструкції для адміністраторів).
  - Підготовка посібника користувача для роботи з особистим кабінетом.
  - Проведення навчання для адміністраторів щодо оновлення контенту через CMS.

9. Підтримка та вдосконалення:

- Моніторинг роботи сайту після запуску (аналіз логів, відгуків користувачів).
- Внесення оновлень на основі зворотного зв'язку (наприклад, додавання нових функцій до кабінету).
- Забезпечення регулярного оновлення безпеки та оптимізації продуктивності.

Ці етапи забезпечать створення якісної інформаційної підсистеми кабінету користувача, яка усуне недоліки аналогів, підвищить зручність для клієнтів і сприятиме розвитку бізнесу провайдера SkyNet.

Висновки до розділу 1

У першому розділі проведено аналіз предметної області, який показав, що сучасного вебсайту для провайдера SkyNet немає, а також виявлено проблеми в існуючих аналогах, зокрема обмежену функціональність їхніх особистих кабінетів. Зробивши аналіз існуючих готових рішень, які б задовольняли наші критерії, таких не виявлено, отже було поставлено за мету спроектувати кабінет користувача для провайдера SkyNet на основі сучасних технологій, використовуючи Java, з урахуванням потреб цільової аудиторії. Аналіз аналогів виявив їхні недоліки, що враховано при розробці концепції SkyNet.

Сформульовано вимоги до сайту, які включають адаптивний дизайн, зручну навігацію, інтерактивні елементи (онлайн-чат, форми), безпечну обробку даних і легке оновлення контенту через CMS. Постановка завдання визначила ключові аспекти створення зрозумілого інтерфейсу, інформативного контенту та привабливого вигляду, що відповідає стандартам веб-розробки.

Засоби розробки:

Java: Використовується для створення безпечного бекенду кабінету, завдяки фреймворкам Spring і Hibernate.



Spring Boot: Для спрощення розробки, автоконфігурації та вбудованого сервера Tomcat.

Spring Security: Для реалізації автентифікації та авторизації користувачів.

MariaDB: Для зберігання даних користувачів із підтримкою JDBC і управління через HeidiSQL.

Bootstrap: Для створення адаптивного фронтенду з готовими компонентами.

Thymeleaf: Для інтеграції HTML-шаблонів із бекендом.

IntelliJ IDEA: Для розробки з інтеграцією Maven і GitHub.

Для реалізації проєкту я буду використовувати Java з екосистемою Spring (Spring Boot, Spring Security, Hibernate) для створення надійного бекенду, MariaDB для управління базами даних, а також Bootstrap і Thymeleaf для адаптивного та зручного фронтенду, що забезпечить ефективну розробку кабінету користувача SkyNet.

## **РОЗДІЛ 2. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ КАБІНЕТУ КОРИСТУВАЧА**

### **2.1 Загальносистемні рішення**

Діаграма Use Case (діаграма варіантів використання) — це інструмент моделювання в UML, який ілюструє функціональні можливості системи з погляду взаємодії користувачів (акторів) із нею. Вона допомагає визначити, які функції має виконувати система і хто їх використовуватиме. Діаграма використовується на етапі проектування для формалізації вимог та забезпечення відповідності системи очікуванням користувачів [1].

Під час проектування інформаційної підсистеми кабінету користувача важливо сформулювати загальну логіку взаємодії користувачів із системою. Для цього застосовується діаграма варіантів використання (Use Case diagram) - один із ключових інструментів моделювання в UML, який демонструє, які функції система виконує з точки зору зовнішніх учасників (акторів). Вона допомагає візуалізувати функціональні вимоги до системи, взаємодії між користувачами та внутрішніми компонентами, а також окреслити межі системи [2].

У контексті розробки кабінету користувача інтернет-провайдера SkyNet було визначено три основні типи акторів:

Клієнт — основний користувач системи, який реєструється, входить до особистого кабінету, переглядає тарифи, подає заявки, здійснює оплату послуг та звертається до техпідтримки.

Адміністратор — керує контентом сайту (тарифні плани, акції), обробляє заявки користувачів, оновлює інформацію в базі даних.

Технічна підтримка — відповідає на запити клієнтів через форму зворотного зв'язку або месенджери.

На рисунку 2.1 зображено діаграму, яка відображає ключові функції для кожного з учасників системи.

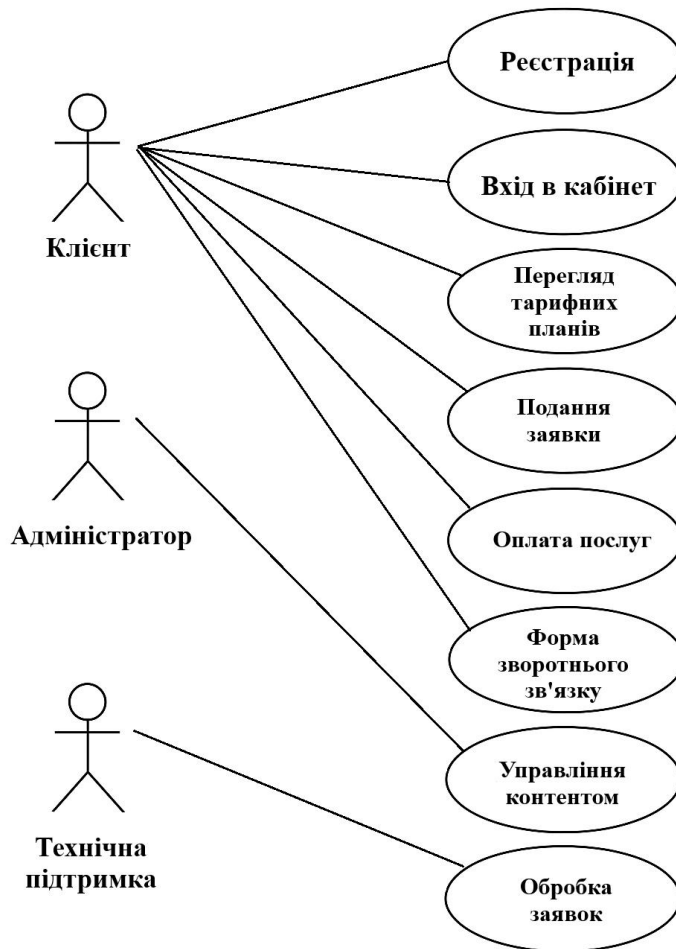


Рисунок 2.1 – Діаграма Use Case

## 2.2 Концептуальна модель

Для ефективної реалізації проєкту інформаційної підсистеми кабінету користувача інтернет-провайдера SkyNet необхідно ретельно визначити всі види даних, які будуть використовуватися та оброблятися в системі, а саме:

Дані про користувачів необхідні для ідентифікації та взаємодії з клієнтами, включаючи персональні дані (ПІБ, дата народження), контактну інформацію (телефон, електронна пошта), дані для автентифікації (логін, пароль) та статус (фізична чи юридична особа).

Інформація про послуги включає перелік доступних тарифних планів (швидкість інтернету, обсяг даних, ціна), додаткові опції (статична IP-адреса, HD-пакети для ТБ, телефонія) та їхні характеристики.

Дані про фінансові операції: Поточний баланс, історія платежів (дата, сума, метод оплати), автоматичні платежі, кешбек та замовлення рахунків для юридичних осіб.

Інформація про технічне обслуговування: Статус підключення, MAC-адреса, історія звернень до техпідтримки, дані про перерви в обслуговуванні чи переїзд.

Ці дані об'єднані в процесі моделювання даних, а саме створення моделі для організації даних, які будуть зберігатися в базі даних. Існує три основні стилі моделювання даних: концептуальні, логічні і фізичні моделі даних, кожна з яких має своє призначення. Моделі даних використовуються для представлення даних і їх зберігання в БД, а також для встановлення зв'язків між елементами даних.

Концептуальна модель даних: основною метою є визначення того, що система містить. Це високорівнева модель, яка використовується для представлення основних бізнес ідей і правил, без детальної технічної реалізації. Вона включає сутності (об'єкти, які важливі для бізнесу), їх атрибути (характеристики або властивості сутностей) та взаємозв'язки між сутностями [3].

На основі вище вказаної інформації було розроблено концептуальну модель, яка представлена на рисунку 2.2.

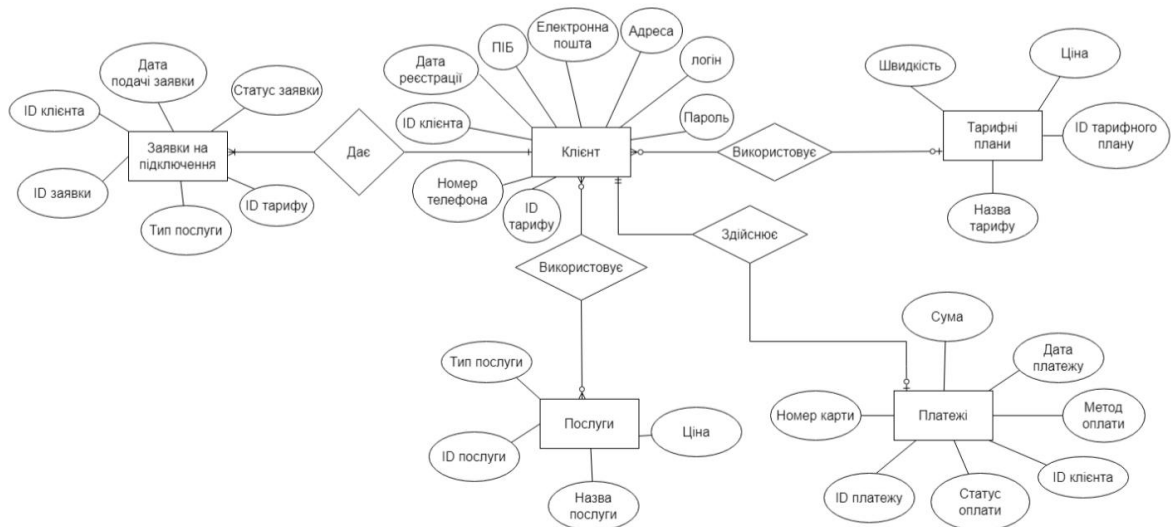


Рисунок 2.2 – Концептуальна модель

Всі сутності атрибути та зв'язки представлені у таблиці 2.1

Таблиця 2.1 – Концептуальна таблиця інтернет-провайдера

| Сутність      | Атрибути             |
|---------------|----------------------|
| Послуги       | Id_services          |
|               | Name_services        |
|               | price                |
| Тарифні плани | id                   |
|               | name                 |
|               | cost                 |
|               | speed                |
| Клієнти       | id                   |
|               | Email                |
|               | Date_register        |
|               | Adress               |
|               | Number_phone         |
|               | Id_tariff_plans      |
|               | login                |
|               | First_name           |
|               | Second_name          |
|               | password             |
| Заявки        | id_zayavki           |
|               | id_users             |
|               | Date_podachi_zayavki |
|               | status_zayavki       |
|               | id_tariff_plans      |

|         |                |
|---------|----------------|
| Платежі | id_payments    |
|         | status_pay     |
|         | id_users       |
|         | date_payments  |
|         | Summa          |
|         | payment_method |
|         | number_card    |

### 2.3 Діаграма послідовності

Діаграма послідовності (Sequence Diagram) використовується для опису динамічної взаємодії між об'єктами в межах одного сценарію використання системи. Вона показує, які повідомлення надсилаються між об'єктами, в якому порядку та як ці повідомлення реалізують функціональність системи [4].

У даному випадку було змодельовано процес подачі заявки на підключення інтернету користувачем через особистий кабінет інтернет-провайдера SkyNet. Діаграма послідовності дозволяє наочно побачити, як система реагує на дії користувача та яким чином обробляється заявка.

На рисунку 2.3 зображено діаграму послідовності, яка демонструє взаємодію між основними компонентами системи:

Користувач (Client) — ініціює подачу заявки;

Інтерфейс (UI) — приймає введені користувачем дані;

Контролер (Controller) — обробляє запит і викликає відповідні методи сервісу;

Сервіс (Service) — здійснює перевірку даних і передає їх у базу;

База даних (Database) — зберігає інформацію про заявку.

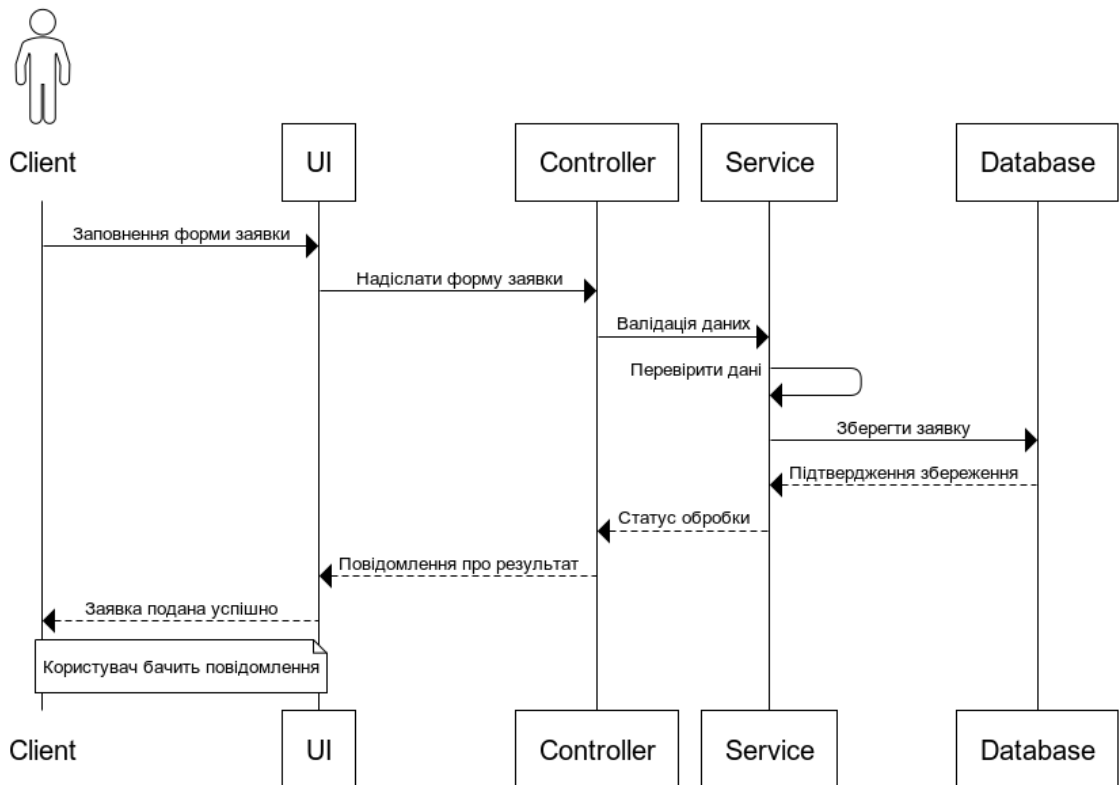


Рисунок 2.3 - Діаграма послідовності

Опис процесу:

1. Користувач вводить дані заявки та натискає кнопку “Підключити”.
2. Дані надсилаються з інтерфейсу на сервер у контролер.
3. Контролер передає їх у сервісний клас.
4. Сервіс перевіряє коректність введеної інформації.
5. У разі успішної перевірки дані записуються у базу даних.
6. База даних повертає відповідь про успішне збереження.
7. Система повідомляє користувача про успішну реєстрацію заявки.

Такий підхід дозволяє спроектувати чітку взаємодію між компонентами системи та переконатися, що реалізація відповідає логіці функціонування, заданій у варіантах використання.

## 2.4 Логічна модель

Логічна модель даних визначає структуру елементів даних і встановлює зв'язки між ними, незалежно від конкретної системи управління базами даних (СУБД). Вона є більш детальною порівняно з концептуальною моделлю і зазвичай створюється архітекторами даних та бізнес-аналітиками. Логічна модель включає конкретні типи даних, зв'язки між таблицями, цілісні правила та інші елементи, які допомагають у визначенні точного способу організації даних [5].

Логічна модель - це більш детальне представлення даних, яке представлено в таблиці 2.2

Таблиця 2.2 – Логічна модель

| Назва таблиці | Атрибут          | Опис даних                                                        |
|---------------|------------------|-------------------------------------------------------------------|
| Клієнт        | ID (PK)          | Унікальний ідентифікатор клієнта                                  |
|               | ПІБ              | Прізвище, ім'я та по батькові клієнта                             |
|               | Електронна пошта | Email-адреса клієнта                                              |
|               | Адреса           | Адреса проживання або підключення клієнта                         |
|               | Номер телефону   | Контактний номер телефону клієнта                                 |
|               | Дата реєстрації  | Дата створення облікового запису клієнта                          |
| Послуги       | ID послуги (PK)  | Унікальний ідентифікатор послуги                                  |
|               | Назва послуги    | Назва послуги, яку надає провайдер                                |
|               | Тип послуги      | Категорія або вид послуги (наприклад, інтернет, телебачення тощо) |
|               | Ціна             | Вартість даної послуги                                            |
|               | ID тарифу (PK)   | Унікальний ідентифікатор тарифного плану                          |
|               | Ім'я             | Назва тарифного плану                                             |



|                       |                    |                                                                     |
|-----------------------|--------------------|---------------------------------------------------------------------|
| Тарифні плани         | Швидкість          | Швидкість надання послуги (наприклад, Мбіт/с)                       |
|                       | Ціна               | Вартість тарифного плану                                            |
| Заявки на підключення | ID заявки (PK)     | Унікальний ідентифікатор заявки на підключення                      |
|                       | ID клієнта         | Зовнішній ключ до клієнта, який подав заявку                        |
|                       | Тип послуги        | Тип послуги, яку хоче підключити клієнт                             |
|                       | Дата подачі заявки | Дата, коли клієнт подав заявку                                      |
|                       | Статус заявки      | Поточний статус заявки (новий, в обробці, виконано, відхилено тощо) |
|                       | ID тарифу          | Зовнішній ключ до вибраного тарифу                                  |
| Платежі               | ID (PK)            | Унікальний ідентифікатор платежу                                    |
|                       | ID клієнта         | Зовнішній ключ до клієнта, що здійснив платіж                       |
|                       | Сума               | Сума оплати                                                         |
|                       | Дата платежу       | Дата, коли був здійснений платіж                                    |
|                       | Статус платежу     | Статус платежу (успішно, неуспішно, очікується)                     |
|                       | Метод платежу      | Спосіб оплати (картка, готівка, онлайн і т.д.)                      |
|                       | Номер картки       | Номер банківської картки, з якої здійснено платіж                   |

Створена логічна модель представлена на рисунку 2.4

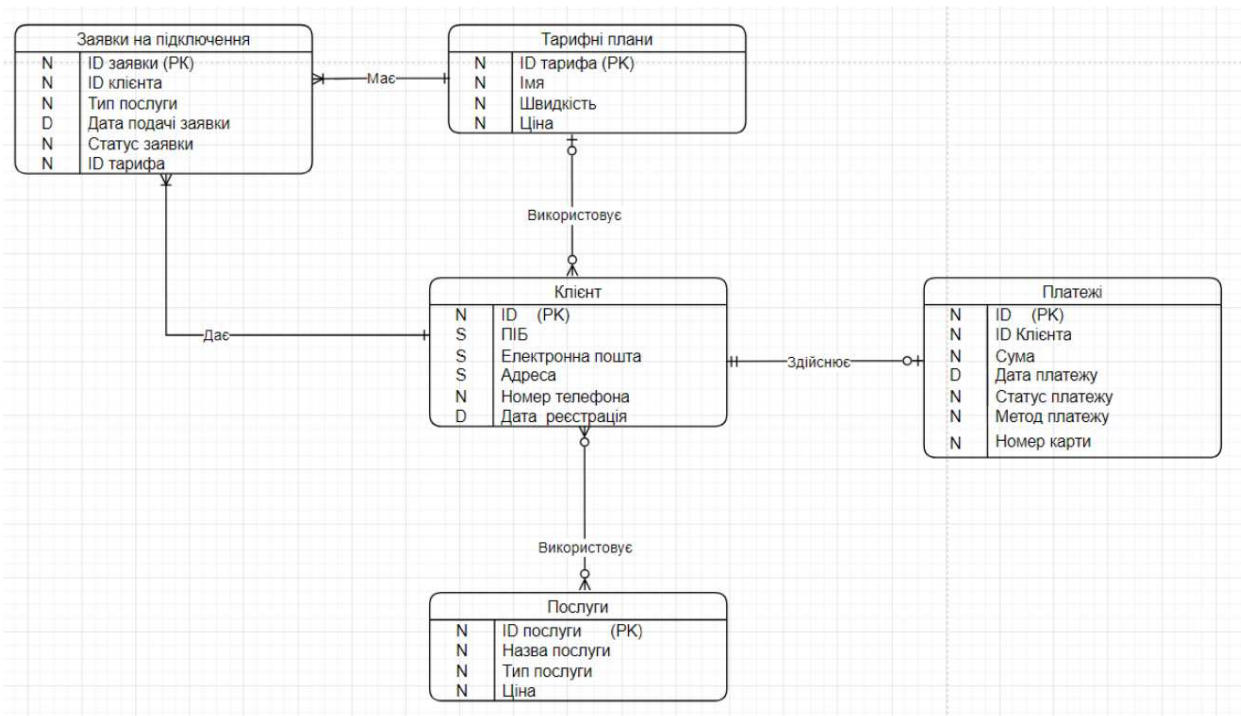


Рисунок 2.4 – Логічна модель

## 2.5 Фізична модель

Розробка фізичної моделі даних, де вона буде описувати базу даних у вигляді конкретних технічних характеристик, а саме нормалізувати, тобто зробити декомпозицію попередніх відношень бази даних на більш прості відношення.

Детальні схеми всіх відношень бази даних з переліком атрибутів, їх типів і розмірів наведені у таблицях 2.3-2.7.

Таблиця 2.3 – Клієнт.

| Назва в БД | Атрибути         | Тип  | Розмір |
|------------|------------------|------|--------|
| id_users   | Id клієнта       | INT  | 11     |
| name       | ПІБ              | TEXT |        |
| Email      | Електронна пошта | TEXT |        |

|               |                |          |  |
|---------------|----------------|----------|--|
| password      | Пароль         | TEXT     |  |
| Adress        | Адреса         | TEXT     |  |
| Number_phone  | Номер телефона | TEXT     |  |
| Date_register | Дата створення | DATETIME |  |

Таблиця 2.4 – Тарифні плани.

| Назва в БД | Атрибути        | Тип    | Розмір |
|------------|-----------------|--------|--------|
| id         | Id тарифа       | INT    | 11     |
| name       | Ім'я            | TEXT   |        |
| cost       | Процента ставка | DOUBLE |        |
| speed      | Швидкість       | INT    | 11     |

Таблиця 2.5 – Заявки на підключення.

| Назва в БД           | Атрибути           | Тип      | Розмір |
|----------------------|--------------------|----------|--------|
| id_zayavki           | Id заявки          | INT      | 11     |
| id_users             | Id клієнта         | INT      | 11     |
| Date_podachi_zayavki | Дата               | DATETIME |        |
| status_zayavki       | Статус             |          |        |
| id_tariff_plans      | Id тарифного плану | INT      | 11     |
| type_of_service      | Тип послуги        | INT      | 11     |

Таблиця 2.6 – Платежі.

| Назва в БД | Атрибути | Тип | Розмір |
|------------|----------|-----|--------|
|            |          |     |        |

|                |               |          |    |
|----------------|---------------|----------|----|
| id_payments    | Id платежа    | INT      | 11 |
| id_users       | Id клієнта    | INT      | 11 |
| date_payments  | Дата          | DATETIME |    |
| status_pay     | Статус        |          |    |
| payment_method | Метод платежу | TEXT     |    |
| number_card    | Номер карти   | TEXT     |    |

Таблиця 2.7 – Послуги.

| Назва в БД      | Атрибути      | Тип      | Розмір |
|-----------------|---------------|----------|--------|
| id_services     | Id послуги    | INT      | 11     |
| name            | Назва послуги | INT      | 11     |
| type_of_service | Тип послуги   | DATETIME |        |
| price           | Ціна послуги  | DOUBLE   |        |

Створена фізична модель представлена на рисунку 2.5

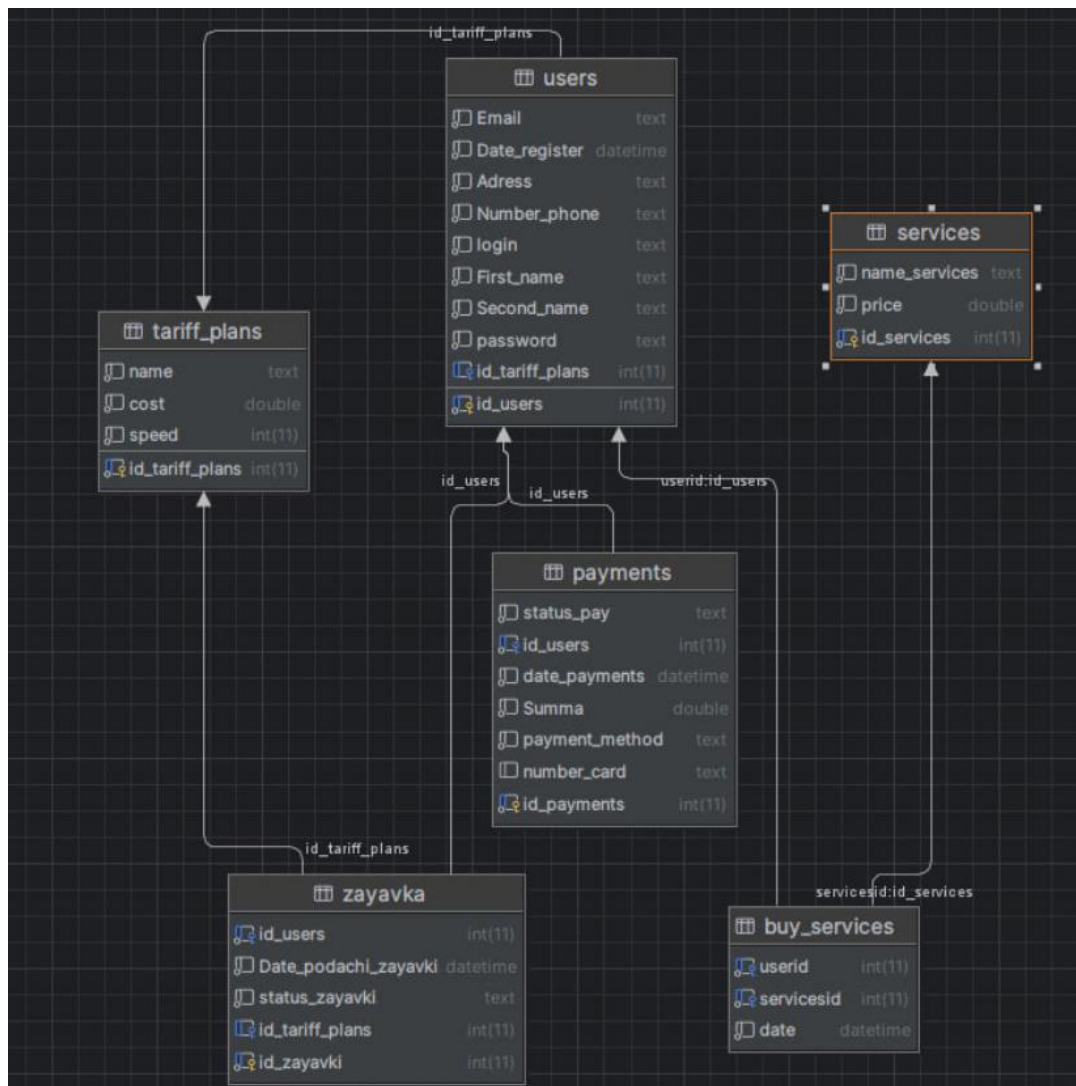


Рисунок 2.5 – Фізична модель

## Висновки до розділу 2

У розділі 2 виконано комплексний аналіз та проектування інформаційної підсистеми особистого кабінету SkyNet. Розроблено діаграму Use Case, яка визначила ключові функції та акторів системи. Діаграма послідовності деталізувала процес подання заявки, ілюструючи взаємодію компонентів. Створено концептуальну, логічну та фізичну моделі даних, нормалізовані до 3NF, що забезпечує цілісність та ефективність бази даних. Обґрунтовано вибір технологій (Java, Spring, MariaDB) та математичних методів (хешування, реляційна алгебра), що гарантують безпеку та продуктивність. Проектування системи повністю відповідає вимогам та готове до реалізації.

## РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ

### 3.1 Діаграма розгортання системи

Діаграма розгортання - це тип UML-діаграми, який використовується для візуалізації фізичної архітектури інформаційної системи. Вона демонструє, як програмні компоненти (модулі, сервіси, бази даних) розміщуються на фізичних вузлах (сервери, клієнтські пристрої тощо) та як між ними здійснюється обмін інформацією [6].

У випадку розробки інформаційної підсистеми кабінету користувача для інтернет-провайдера SkyNet було спроектовано розгортання системи за класичною моделлю «клієнт-сервер».

Основу складають три головні вузли:

Клієнтський пристрій (Client) - пристрій користувача, що взаємодіє із системою через браузер. На ньому відображається інтерфейс (HTML, CSS, JavaScript), згенерований сервером.

Веб-сервер (Application Server) - сервер, на якому розміщено бекенд-систему, реалізовану на Java з використанням фреймворку Spring Boot. Сервер обробляє HTTP-запити користувача, забезпечує авторизацію, бізнес-логіку та обмін даними з базою.

Сервер бази даних (Database Server) - вузол, де розміщується СУБД MariaDB, яка зберігає дані про користувачів, тарифи, заявки та платежі.

Взаємодія між вузлами:

Користувач у браузері надсилає запит до веб-сервера (наприклад, на вхід або перегляд тарифів).

Веб-сервер обробляє запит через контролери, сервіси, перевіряє авторизацію.

За потреби дані запитуються або зберігаються в базу даних.

Отримані результати повертаються користувачу у вигляді HTML-сторінки.

Компоненти діаграми:

Client Device

Браузер (Google Chrome, Firefox, Safari)

Application Server

Spring Boot додаток

Tomcat Embedded

Thymeleaf (генерація HTML)

Spring Security (автентифікація)

Database Server

MariaDB

HeidiSQL для адміністрування

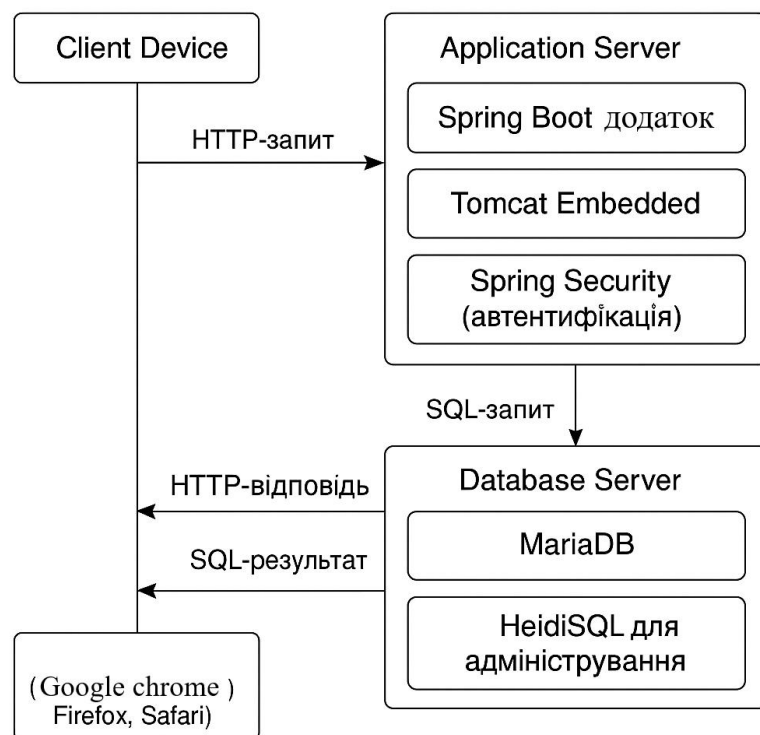


Рисунок 3.1 - Діаграма розгортання

На рисунку 3.1 представлено діаграму розгортання, яка демонструє фізичне розміщення основних компонентів системи на різних вузлах.

### 3.2 Рішення щодо апаратного забезпечення

Апаратне забезпечення — це сукупність фізичних засобів, необхідних для функціонування інформаційної підсистеми. У системі кабінету користувача інтернет-провайдера SkyNet апаратна інфраструктура поділяється на три основні компоненти: серверну частину, клієнтське обладнання та апаратне забезпечення розробника [7].

#### Серверна частина (Production Server)

Сервер забезпечує виконання бекенд-компонентів, обробку запитів клієнтів, зберігання даних у базі даних та взаємодію з клієнтськими пристроями через мережу.

Рекомендовані характеристики серверу:

Процесор (CPU): 4-ядерний Intel Core i5 / AMD Ryzen 5 або вище

Оперативна пам'ять (RAM): мінімум 8 GB (рекомендовано — 16 GB для багатозадачності)

Накопичувач: SSD 120–256 GB для швидкої обробки запитів

Мережева карта: Ethernet 1 Gbps

Операційна система: Ubuntu Server 22.04 LTS або аналогічна Linux-based ОС

Додатково: можливість резервного копіювання даних, стабільне підключення до Інтернету

#### Сервер бази даних

Хоча база даних може бути розгорнута на тому ж сервері, для продуктивності та безпеки доцільно виділити окремий сервер або віртуальну машину.

Вимоги до апаратного забезпечення бази даних MariaDB:

CPU: 2 ядра

RAM: 4–8 GB

HDD: 50–100 GB SSD

ОС: Ubuntu Server або Windows Server



Клієнтські пристрої

Користувачі взаємодіють із системою через браузер, тому спеціалізовані вимоги відсутні. Система повинна коректно працювати на більшості сучасних пристроїв.

Мінімальні вимоги до пристрою користувача:

Процесор: 2-ядерний або вище

ОЗП: від 2 GB

ОС: Windows 10, macOS, Linux, Android або iOS

Браузери: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari (із підтримкою HTML5 та CSS3)

Дисплей: Рекомендована роздільна здатність — 1280×720 або вище

4. Апаратне забезпечення розробника

У процесі розробки було використано стандартний ноутбук з такими параметрами:

Процесор: Intel Core i5-1035G1, 1.0 GHz (4 ядра)

ОЗП: 8 GB

Накопичувач: SSD 256 GB

ОС: Windows 11

ПЗ: IntelliJ IDEA Ultimate, XAMPP, HeidiSQL, браузери для тестування

Таким чином, апаратне забезпечення проєкту SkyNet є відносно недорогим і може бути реалізоване навіть на бюджетному обладнанні. Рекомендована серверна конфігурація дозволяє забезпечити стабільну роботу системи при одночасному обслуговуванні сотень користувачів.

### 3.3 Рішення щодо інформаційного забезпечення

У наш час існує дуже багато мов для створення сайтів, опишемо найпопулярніші, такі як: HTML, CSS, JavaScript, SQL, Java .

## Java

Java є однією з найпопулярніших і найбільш широко використовуваних мов програмування для створення веб-застосунків [8].

Вибір Java для цієї курсової роботи обумовлений наступними причинами:

### 1) Розвинена екосистема для веб-розробки

Java пропонує широкий спектр інструментів і фреймворків для створення веб-застосунків. Основними серед них є:

- Spring Framework: забезпечує зручну роботу з веб-сервісами, інтеграцію з базами даних, обробку запитів і забезпечення безпеки.
- Java EE (Jakarta EE): набір специфікацій для корпоративного рівня, включаючи сервіслети, JSP, WebSocket та інше.
- Hibernate: фреймворк для роботи з базами даних через ORM (Object-Relational Mapping), що значно спрощує взаємодію з реляційними базами даних.

### 2) Широкий набір бібліотек

Java має величезний набір бібліотек для роботи з мережею, JSON/XML, інтеграції API, роботи з базами даних і реалізації інших функцій, що значно прискорює розробку.

### 3) Безпека

Java має вбудовані механізми безпеки, які дозволяють створювати надійні веб-застосунки. Наприклад, Java забезпечує захист від таких загроз, як SQL-ін'єкції, завдяки інтеграції з ORM-фреймворками. Крім того, Java дозволяє легко впроваджувати механізми автентифікації, авторизації та шифрування.

## Середовище розробки

IntelliJ IDEA - це потужне інтегроване середовище розробки (IDE), розроблене компанією JetBrains. Воно призначене для написання програм на Java, Kotlin, Groovy, Scala, Python, JavaScript, TypeScript, HTML, CSS та інших мовах. IntelliJ IDEA має два основних видання: Community Edition

(безкоштовне) та Ultimate Edition (комерційне), кожне з яких має свої особливості та функціонал.

### Основні переваги IntelliJ IDEA

IntelliJ IDEA аналізує код у реальному часі, забезпечуючи автодоповнення, підсвічування помилок, швидкі виправлення, рефакторинг і підтримку шаблонів коду. IDE надає можливість швидкого переходу між файлами, класами, методами або змінними, що значно прискорює розробку. Підтримуються системи Git, GitHub, має засоби для роботи з такими платформами, як Spring, Hibernate, Java EE, Jakarta EE, Grails, Vaadin, Angular, React та інші.

В цьому проєкті IntelliJ IDEA використовуватиметься з системою зборки проєктів Maven, яка значно спростить підключення бібліотек та плагінів. Інтеграція IDE з GitHub дозволить мені зручно контролювати версії програми та зміни в ній [9].

### Додаткові інструменти розробки:

XAMPP Control panel – дозволяє легко запускати і вимикати сервер за бажанням користувача, на відміну від інших середовищ, які запускаються або разом з комп'ютером, або вимагають запуску через консоль.

HeidiSQL – програма зі зручним графічним інтерфейсом, дозволяє зручно створювати та редагувати таблиці в базі, зручний доступ до даних в них [10].

### Фреймворки та бібліотеки

Spring Initializr - інструмент для швидкого створення проєктів Spring Boot.

Spring Boot - спрощує розробку Java застосунків, надаючи вбудований Tomcat та автоконфігурацію.

Spring Data JPA - надає абстракцію для роботи з ORM (Object Relational Mapping), що прискорює операції з базами даних.

Spring Security - інструмент для реалізації аутентифікації та авторизації в веб-застосунку.

Hibernate - забезпечує автоматичну відповідність між об'єктами Java та таблицями базами даних, виконує автоматичне керування SQL-запитами для збереження, оновлення та видалення даних.

Thymeleaf - шаблонізатор, що дозволяє просто та зручно пов'язувати HTML з даними з бекенду [11].

jQuery - бібліотека JavaScript для швидкого маніпулювання DOM та створення інтерактивних елементів.

Lombok - значно скорочує кількість шаблонного коду, необхідного для розробки Java-застосунків, дозволяє автоматично генерувати такі елементи, як геттери, сеттери, конструктори, методи toString, equals, hashCode тощо

Sql

MariaDB є форком MySQL, що забезпечує сумісність з MySQL на рівні API, SQL-запитів і структур даних. Це дозволяє використовувати численні інструменти та бібліотеки, розроблені для MySQL, такі як Hibernate, JPA або Spring Data.

Зручність інтеграції з Java - Завдяки JDBC (Java Database Connectivity) драйверам для MariaDB, інтеграція з веб-застосунком, написаним на Java, є простою та ефективною. Більше того, MariaDB добре працює у фреймворках Spring Boot та Hibernate, які використовуються для розробки сучасних веб-додатків.

Легкість адміністрування - Для MariaDB доступні інтуїтивно зрозумілі інструменти адміністрування, такі як phpMyAdmin, DBeaver та HeidiSQL, які спрощують налаштування та управління базою даних навіть для новачків.

#### Фронтенд-інструменти

Окрім базових HTML, CSS в проєкті буде використано Bootstrap - який вже готові компоненти для створення веб-інтерфейсів прямо «із коробки», кнопки, форми, навігаційні панелі, таблиці, модальні вікна – все це вже матиме гарний дизайн та зекономить час.

Плагіни та розширення

JPA Buddy - спрощує роботу з Hibernate, Spring Data JPA та іншими інструментами ORM.

The Database Tools and SQL - надає засоби для керування базами даних, виконання запитів та створення діаграм.

IntelliJ Lombok Plugin - забезпечує коректну роботу з Lombok в IDE.

Thymeleaf Plugin - підтримує шаблони Thymeleaf у IntelliJ IDEA та полегшує роботу з XML/XHTML/HTML5.

Spring Initializr - забезпечує коректну роботу з Spring Initializr в IDE

Spring Boot - забезпечує коректну роботу з Spring Boot в IDE

Spring Data - забезпечує коректну роботу з Spring Data в IDE

Spring Security - забезпечує коректну роботу з Spring Security в IDE

### 3.4 Екранні форми розробки

На початку розроблення проєкту інформаційної підсистеми кабінету користувача для інтернет-провайдера SkyNet було виконано проєктування структури сайту та його функціоналу з використанням сучасних інструментів розробки. Розробка інтерфейсу та бекенду здійснювалася з урахуванням вимог до адаптивності, інтерактивності та безпеки, що відображено в подальших ескізах і реалізації сторінок сайту.

Розробка інтерфейсу базувалася на визначених вимогах до зручності та функціональності. Нижче представлено ескізи та опис ключових сторінок сайту SkyNet, які реалізують основні можливості для користувачів і адміністраторів.

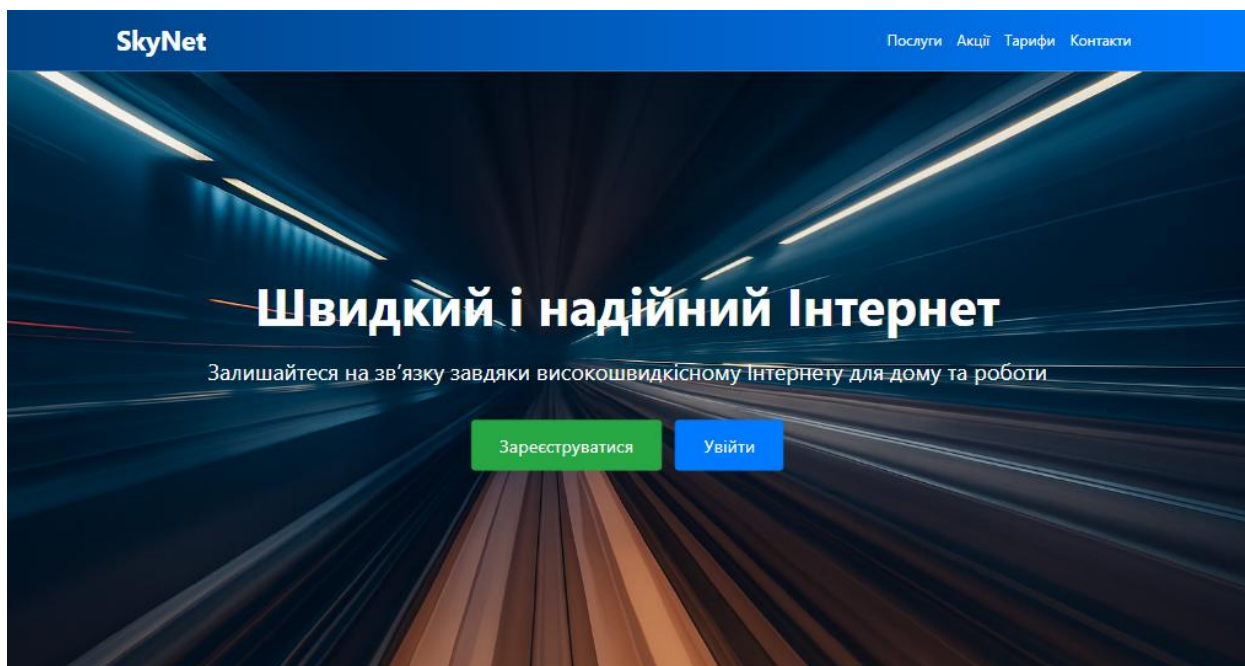


Рисунок 3.2 – Головна сторінка

На головній сторінці розміщено меню з основними розділами (Послуги, Акції, Контакти) та кнопки для реєстрації чи входу до акаунту, що забезпечує швидкий доступ до сервісів.

Головна сторінка надає вичерпну інформацію про інтернет-послуги, які пропонує компанія. За допомогою інтерактивних елементів, таких як слайдер з акціями, блок переваг, тарифні плани та форма зворотного зв'язку, користувачі можуть швидко ознайомитися з актуальними пропозиціями, ключовими характеристиками сервісу та способами зв'язку з оператором.

Такий підхід забезпечує зручну навігацію, підвищує зацікавленість відвідувачів і сприяє прийняттю позитивного рішення щодо підключення послуг, що представлено на рисунках 3.3, 3.4, 3.5.

## Тарифні плани

|                                                                                                                                 |                                                                                                                                           |                                                                                                                                               |
|---------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Базовий</b><br><b>130 грн/міс</b><br>Швидкість до 100 Мбіт/с<br>100+ телеканалів<br>Безліміт<br><a href="#">Підключитися</a> | <b>Стандартний</b><br><b>200 грн/міс</b><br>Швидкість до 300 Мбіт/с<br>150+ телеканалів<br>Підтримка 24/7<br><a href="#">Підключитися</a> | <b>Преміум</b><br><b>300 грн/міс</b><br>Швидкість до 1000 Мбіт/с<br>200+ телеканалів<br>Персональний менеджер<br><a href="#">Підключитися</a> |
|---------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|

Рисунок 3.3 – Блок тарифних планів

## Акції

**ПОДАРУНОК  
ПРИ ПІДКЛЮЧЕННІ**

Отримайте роутер  
у подарунок  
при підключенні  
інтернету на рік!



Рисунок 3.4 – Блок акцій

## Наші переваги

|                                                                                                                                                                     |                                                                                                                                                                   |                                                                                                                                                                   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <br><b>Швидкість</b><br>Блискавичний Інтернет до 1 Гбіт/с для всіх ваших потреб. | <br><b>Безпека</b><br>Шифрування та захист особистих даних на найвищому рівні. | <br><b>Підтримка 24/7</b><br>Наша служба підтримки завжди на зв'язку з вами. |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Рисунок 3.5 – Переваги інтернет провайдера

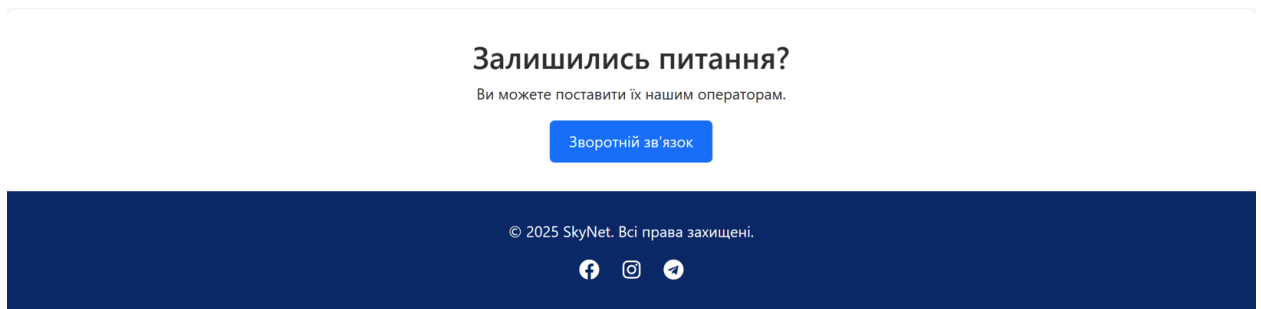


Рисунок 3.6 – Зворотній зв'язок

Сторінка форми зворотного зв'язку де користувач може зв'язатися з підтримкою та вирішити свої проблеми, заповнивши просту форму з полями для імені, email та опису питання, а потім натиснувши кнопку "Відправити". Виконана у світлих тонах із чітким заголовком, вона проста і зручна для швидкого звернення за допомогою.

Рисунок 3.7 – Форма зворотнього зв'язку



### Висновки до розділу 3

У розділі 3 описано реалізацію інформаційної підсистеми «SkyNet». Розроблено діаграму розгортання, що ілюструє фізичну архітектуру системи. Описано рішення щодо інформаційного забезпечення, включаючи обробку даних з використанням Spring Data JPA та MariaDB. Визначено вимоги до апаратного забезпечення, що забезпечують стабільність та масштабованість. Представлені ключові екранні форми (головна сторінка, особистий кабінет, форма зворотного зв'язку), що демонструють зручність та адаптивність інтерфейсу. Реалізація системи повністю відповідає вимогам, викладеним у розділі 2, та готова до впровадження

## РОЗДІЛ 4. ОХОРОНА ПРАЦІ

Розробка підсистеми інтернет-провайдера SkyNet передбачає інтенсивну інтелектуальну працю програмістів, пов'язану з тривалим використанням комп'ютерної техніки. Відповідно до Закону України "Про охорону праці" та нормативних актів, роботодавець зобов'язаний забезпечити безпечні умови праці, мінімізувати вплив шкідливих факторів та запобігати професійним захворюванням.

Основними аспектами охорони праці в даному контексті є:

- Дотримання санітарних норм роботи з екранними пристроями (ДСанПіН 3.3.2.007-98) [12].
- Організація ергономічного робочого місця.
- Забезпечення психоемоційного комфорту та запобігання професійному вигоранню.
- Регулярні медичні огляди працівників (Наказ МОЗ №246 від 21.05.2007) [13].

### 4.1 Шкідливі та небезпечні фактори праці програміста

Праця програміста супроводжується низкою шкідливих факторів, які можуть негативно впливати на здоров'я:

Фізіологічні:

- Навантаження на зір через тривалу роботу з моніторами може викликати синдром комп'ютерного зору, що проявляється втомою очей, головним болем, сухістю та почервонінням.
- Порушення постави та м'язово-скелетні розлади (остеохондроз, синдром зап'ястного каналу).
- Вплив електромагнітного випромінювання від комп'ютерів також становить потенційний ризик.
- Психологічні:

- Високий рівень стресу, тиск дедлайнів, соціальна ізоляція та необхідність постійного оновлення знань можуть сприяти хронічному стресу, професійному вигоранню чи депресії.

- Організаційні:
- Порухення режиму праці та неналежна організація робочого місця

#### 4.2 Ергономічні та санітарні вимоги до робочого місця

Робочі місця програмістів повинні відповідати вимогам:

Простір: Мінімум 4.5 м<sup>2</sup> на працівника (згідно з ДСанПіН 3.3.2.007-98).

Обладнання:

Регульований стіл (висота 680-800 мм)

Крісло має бути регульованим, обертальним, із підтримкою спини та хребта. За потреби передбачається підставка для ніг.

Монітор з антибликовим покриттям повинен бути розташований перпендикулярно до лінії зору на відстані 600-700 мм від очей.

Клавіатура з матовою поверхнею, має розміщуватися на відстані 100-300 мм від краю столу з регульованим нахилом від 5 до 15 градусів

Мікроклімат:

Температура: 22-24°C, вологість 40-60% (ДСН 3.3.6.042-99) [14].

Вентиляція та відсутність шуму понад 50 дБ (ДСН 3.3.6.037-99) [15].

Освітлення:

Освітлення має створювати належний контраст між екраном та оточуючим середовищем, відповідаючи вимогам ДСанПіН 3.3.2.007-98 [12]. Вікна повинні бути обладнані регульованими жалюзі чи шторами, а комп'ютери розміщені так, щоб світло падало на екрани з півночі чи північного сходу. Також важлива відсутність прямих джерел світла, що створюють відблиски. Рівень освітленості має складати 300-500 лк.

#### 4.3 Нормативні вимоги до організації праці

Режим роботи: Перерви кожні 2 години тривалістю 10-15 хв. для гімнастики очей та розминки (згідно з ДСанПіН 3.3.2.007-98).

Обмеження безперервної роботи з монітором до 6 годин на день.

Навчання: Інструктаж з техніки безпеки під час роботи з електрообладнанням.

Тренінги з ергономіки та профілактики професійних захворювань.

Медичні огляди: Попередній та періодичні огляди (1 раз на 2 роки) для виявлення порушень зору, опорно-рухового апарату.

#### 4.4 Щоденні процедури та заборони

Очищення: Щоденно перед початком роботи екранні пристрої необхідно очищати від пилу та інших забруднень. Приміщення потребують щоденного вологого прибирання та регулярного провітрювання.

Відключення: Після завершення роботи пристрої слід відключати від електромережі. У разі аварійної ситуації необхідно негайно відключити пристрій.

Заборони:

- Виконувати технічне обслуговування, ремонт чи налагодження екранних пристроїв на робочому місці під час роботи.
- Відключати захисні пристрої чи вносити несанкціоновані зміни до конструкції пристроїв.
- Працювати з пристроями, що мають нехарактерні сигнали, нестабільне зображення чи інші несправності.
- Розташовувати комп'ютери в підвальних приміщеннях чи поблизу джерел тепла (менше 1 метра).
- Використовувати несправні монітори чи матричні принтери без антивібраційних чохлів.

#### 4.5 Покарання за невідповідність

Відповідно до статті 43 Закону України "Про охорону праці", порушення законодавства про охорону праці тягне за собою накладення штрафів. Максимальний розмір штрафу не може перевищувати 5% середньомісячного фонду заробітної плати за попередній рік. Несплата штрафу призводить до нарахування пені у розмірі 120% річних облікової ставки Національного банку України за кожен день прострочення. Сплата штрафу не звільняє від усунення виявлених порушень.

Дотримання вимог охорони праці є необхідною умовою для створення безпечного та продуктивного робочого. Впровадження ергономічних стандартів, регулярних перерв, медичних оглядів та належного навчання дозволяє значно знизити ризик професійних захворювань і підвищити ефективність роботи команди розробників. Ці заходи не лише відповідають законодавчим вимогам, але й сприяють загальному добробуту працівників [16].

#### Висновки до розділу 4

У даному розділі було розглянуто основні аспекти охорони праці, що мають важливе значення для забезпечення безпечних і комфортних умов роботи програмістів. Проаналізовано шкідливі та небезпечні фактори, з якими стикаються фахівці при тривалій роботі з комп'ютером: зорове навантаження, порушення постави, психоемоційне вигорання. Надано рекомендації щодо ергономіки робочого місця, мікроклімату, організації режиму праці, що відповідають чинним нормативам і стандартам.

Також у розділі визначено санітарно-гігієнічні вимоги до робочого простору та технічного оснащення, які сприяють зменшенню негативного впливу комп'ютерного середовища на організм працівника. Реалізація зазначених заходів дозволяє підвищити продуктивність праці, знизити ризики для здоров'я працівників і створити комфортне робоче середовище для реалізації ІТ-проектів.

## **ВИСНОВКИ**

Під час виконання дипломної роботи виконано наступні завдання:

- проаналізовано вебсайти інтернет провайдерів;
- проаналізовано сучасні технології у створенні вебсайтів та досліджено засоби розробки;
- розроблено проєкт сайту інтернет провайдера SkyNet;
- розроблено сайт інтернет провайдера SkyNet;

Розділи розробленого вебсайту: Головна, Акції, Тарифи, Послуги, Контакти.

Створений програмний продукт задовольняє всім вимогам, що були зазначені на етапі постановки завдання.

## ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Компанєєтс М. О. Принципи проектування ефективних веб-сайтів. 2015. 109 с. URL: <http://molodyvcheny.in.ua/files/journal/2015/9/66.pdf> (дата звернення: 12.06.2025).
2. Діаграма прецедентів у моделюванні систем URL: [uk.wikipedia.org/wiki/Діаграма\\_прецедентів](http://uk.wikipedia.org/wiki/Діаграма_прецедентів)
3. Концептуальна модель даних URL: [uk.wikipedia.org/wiki/Концептуальна\\_модель](http://uk.wikipedia.org/wiki/Концептуальна_модель)
4. Діаграми послідовності в UML URL: <https://www.maxzosim.com/sequence-diagrams/>
5. Логічна модель даних у проектуванні баз даних URL: <https://data-life.ua.com/db/lohichna-model-danykh/>
6. Діаграма розгортання в UML URL: <https://www.guru99.com/uk/deployment-diagram-uml-example.html>
7. Основи апаратного забезпечення комп'ютерних систем URL: <https://www.znaiemoinformatyku.org/aparatne-zabezpechennia-osnovy>
8. Використання Java у веб-розробці URL: <https://pnn.com.ua/ua/blog/detail/what-is-java-used-for>
9. Інтегроване середовище розробки IntelliJ IDEA URL: [uk.wikipedia.org/wiki/IntelliJ\\_IDEA](http://uk.wikipedia.org/wiki/IntelliJ_IDEA)
10. HeidiSQL URL: [uk.wikipedia.org/wiki/HeidiSQL](http://uk.wikipedia.org/wiki/HeidiSQL)
11. Daber J. Html, Css, Website development and design. Exemo, 2022.
12. Державні санітарні правила роботи з комп'ютерами URL: <https://zakon.rada.gov.ua/rada/show/v0007282-98#Text>
13. Порядок проведення медичних оглядів працівників URL: <https://zakon.rada.gov.ua/laws/show/z0846-07#Text>
14. Державні санітарні норми мікроклімату виробничих приміщень URL: <https://zakon.rada.gov.ua/rada/show/va042282-99#Text>
15. Державні санітарні норми допустимих рівнів шуму URL: <https://zakon.rada.gov.ua/rada/show/va037282-99#Text>

16. Закон України «Про охорону праці» URL:  
<https://zakon.rada.gov.ua/laws/show/2694-12#Text>



## ДОДАТОК А - ІНСТРУКЦІЯ КОРИСТУВАЧА

При вході на сайт ви потрапляєте на головну сторінку, де можете ознайомитися з детальною інформацією про каршерінг та з його перевагами, який представлено на рисунку А.1.

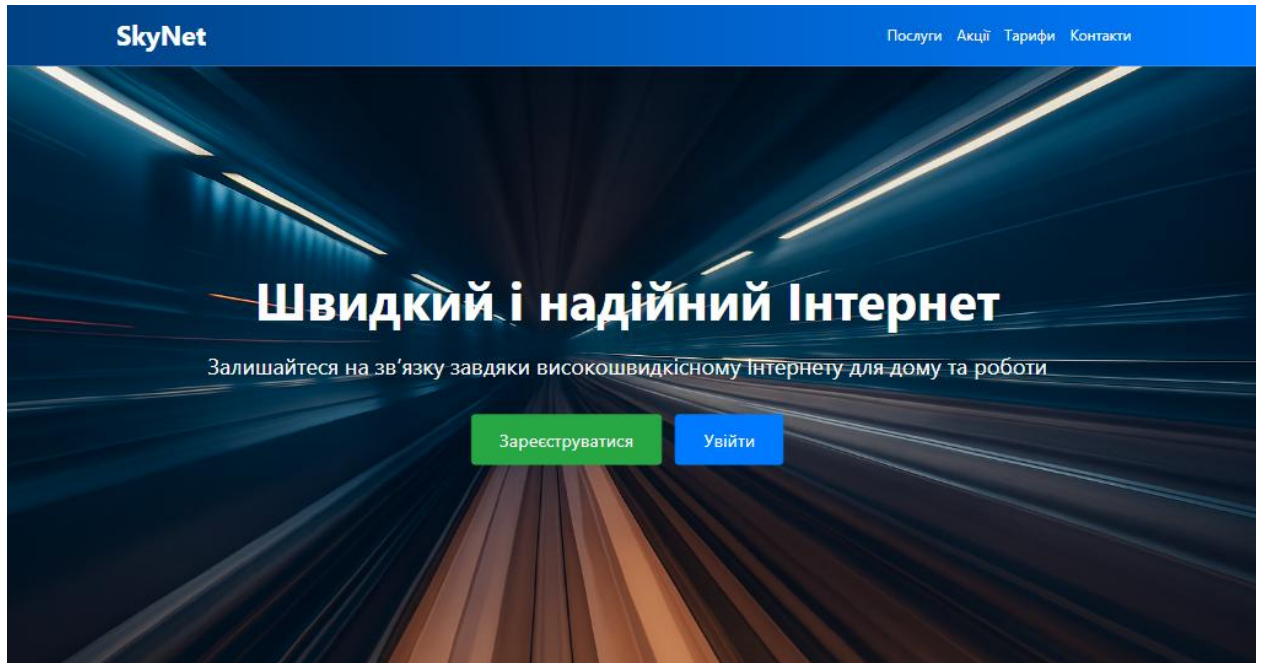


Рисунок А.1 – Головна сторінка

Вгорі сайту розташований навігаційний блок, який дозволяє швидко переходити на потрібну сторінку, що представлено на рисунку А.2.



Рисунок А.2 – Навігаційний блок

Для оформлення замовлення тарифного плану потрібно спочатку зареєструватися, вказавши електронну пошту, ім'я, прізвище, телефон, логін, вгадати пароль і повторити його. Якщо у вас вже є обліковий запис, ви можете написати логін, а також пароль щоб увійти та перейти до вашого профілю користувача. Це все представлено на рисунках А.3, А4.

**Реєстрація**

Ім'я:

Прізвище:

Пошта:

Адреса:

Номер телефону:

Логін:

Пароль:

[Реєстрація](#)

Маєте акаунт? [Логін](#)

[Головна сторінка](#)

Рисунок А.3 – Сторінка реєстрації акаунту

**Вхід**

Логін:

Пароль:

[Вхід](#)

[Головна сторінка](#)

Рисунок А.4 – Вхід до облікового акаунту

Після авторизації ви потрапляєте до особистого кабінету, який забезпечує доступ до основної інформації профілю та функцій керування обліковим записом. У верхній частині сторінки відображається привітання з ім'ям користувача. Далі представлено блоки з особистими даними (електронна пошта, номер телефону, адреса проживання) та можливістю редагування профілю.

Ви маєте змогу переглянути поточний тарифний план, підключення до інтернету та, за потреби, оформити нове замовлення та підключити інтернет натисканням відповідної кнопки. Статус підключення відображається кольоровим індикатором (зелений – підключено, червоний – не підключено).

Функціонал та особистий кабінет користувача представлено на рисунках А.5, А.6, А.7, А.8.

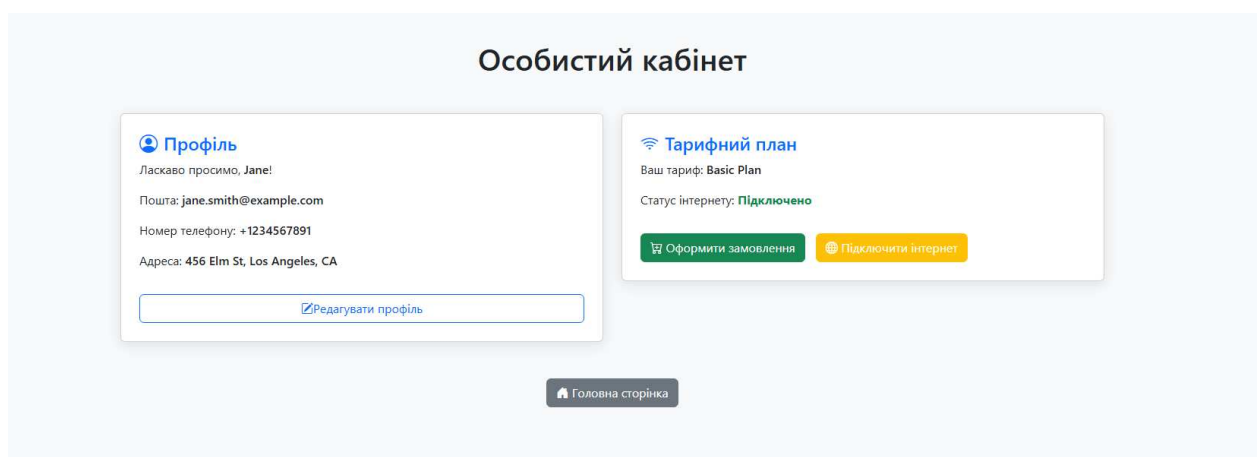
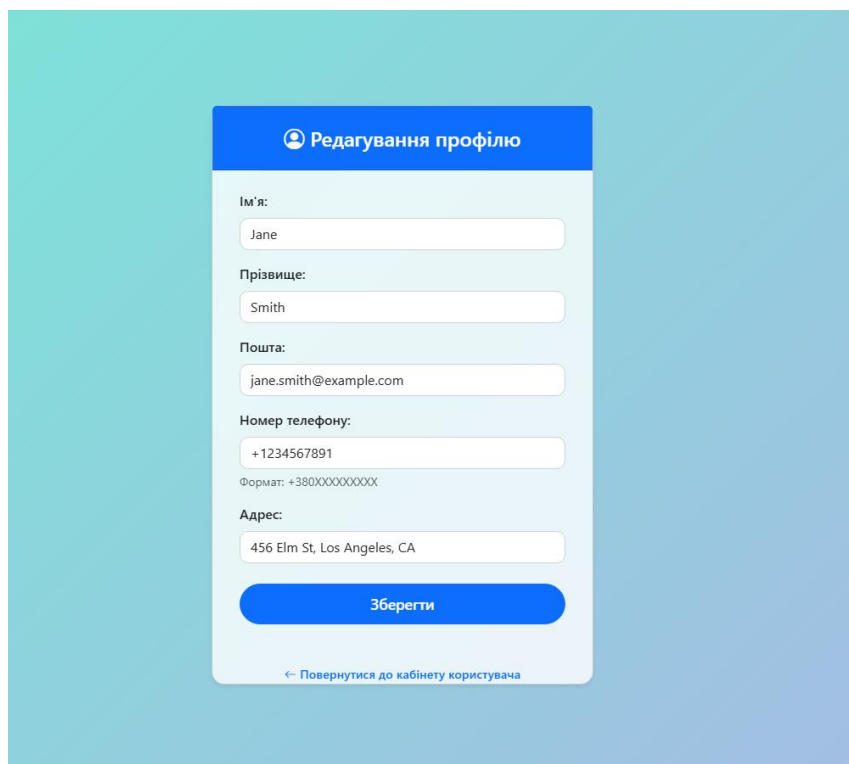


Рисунок А.5 – Профіль користувача



**Редагування профілю**

Ім'я: Jane

Прізвище: Smith

Пошта: jane.smith@example.com

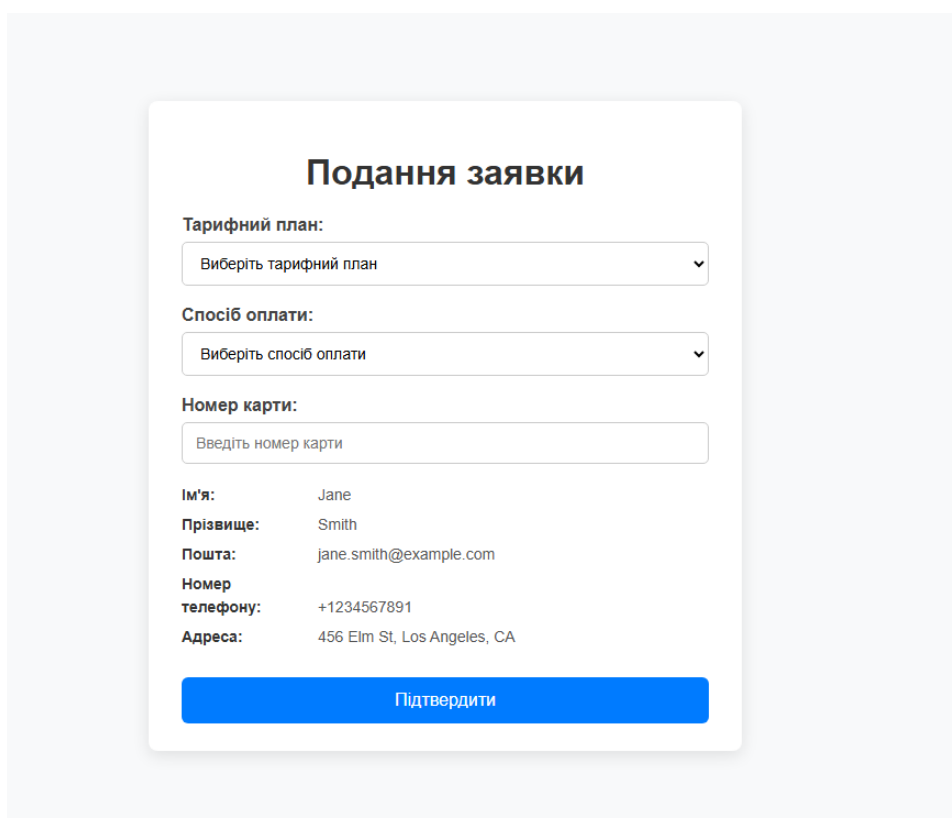
Номер телефону: +1234567891  
Формат: +380XXXXXXXXXX

Адрес: 456 Elm St, Los Angeles, CA

**Зберегти**

[← Повернутися до кабінету користувача](#)

Рисунок А.6 – Редагування профілю



**Подання заявки**

Тарифний план: Виберіть тарифний план

Спосіб оплати: Виберіть спосіб оплати

Номер карти: Введіть номер карти

Ім'я: Jane

Прізвище: Smith

Пошта: jane.smith@example.com

Номер телефону: +1234567891

Адреса: 456 Elm St, Los Angeles, CA

**Підтвердити**

Рисунок А.7 – Оформлення замовлення

# Підключення інтернету

Адреса:

456 Elm St, Los Angeles, CA

Номер телефону:

+1234567891

Ваш тарифний план: [Basic Plan](#)

Підключити

Рисунок А.8 – Підключення інтернету

## ДОДАТОК Б - ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ

### 1. Призначення та мета створення сайту інтернет провайдера

Сайт призначений для забезпечення онлайн-присутності компанії, що надає послуги інтернет-зв'язку. За його допомогою користувачі можуть ознайомитися з інформацією про компанію, дізнатися про доступні тарифні плани, умови підключення, а також оформити заявку на підключення.

Мета створення сайту - розробити зручний і сучасний онлайн-ресурс, який дозволить компанії презентувати себе в Інтернеті, надавати актуальну інформацію про послуги, тарифи, умови підключення, а також забезпечити ефективне спілкування з клієнтами.

### 2. Вимоги до сайту

Функціональні вимоги до сайту SkyNet

Головна сторінка: Відображення інформації про компанію, тарифи, акції; навігаційне меню («Про нас», «Тарифи», «Підключення», «Підтримка», «Контакти»); слайдер акцій; кнопки реєстрації/входу.

Особистий кабінет: Реєстрація/авторизація, перегляд та редагування профілю, управління тарифами, подання заявок, перегляд балансу та платежів.

Тарифи та послуги: Відображення тарифів, підключення послуг.

Зворотній зв'язок: Форма звернень, інтеграція з месенджерами.

Управління контентом через CMS, обробка заявок, оновлення бази даних.

Платежі: Онлайн-оплата, перегляд історії платежів, автоплатежі.

### 3. Нефункціональні вимоги

Адаптивність: Коректне відображення на всіх пристроях та браузерях (Bootstrap).

Безпека: Захист даних (Spring Security), шифрування платежів, захист від SQL-ін'єкцій та XSS.

Продуктивність: Швидка обробка запитів, підтримка сотень користувачів.

Зручність: Інтуїтивна навігація, сучасний UX/UI-дизайн.

Масштабованість: Можливість додавання нових функцій через CMS.

Надійність: Резервне копіювання, стабільне підключення.

#### 4. Технології

Java: Основна мова для серверної логіки, обрана через розвинену екосистему (Spring Framework, Hibernate) та вбудовані механізми безпеки (захист від SQL-ін'єкцій, автентифікація).

HTML, CSS, JavaScript: Для створення інтерфейсу користувача, стилізації та інтерактивності. Використано бібліотеку jQuery для маніпуляцій з DOM.

SQL (MariaDB): Для управління базами даних через JDBC та інтеграцію з Hibernate.

Фреймворки та бібліотеки:

Spring Boot: Спрощує розробку Java-застосунків із вбудованим Tomcat.

Spring Data JPA, Hibernate: Для роботи з базами даних через ORM.

Spring Security: Забезпечує автентифікацію та авторизацію.

Thymeleaf: Для створення шаблонів HTML із прив'язкою до бекенду.

Bootstrap: Для адаптивного дизайну та готових компонентів інтерфейсу.

Lombok: Для зменшення шаблонного коду (геттери, сеттери).

Середовище розробки:

IntelliJ IDEA: IDE з підтримкою автодоповнення, рефакторингу та інтеграції з GitHub і Maven.

XAMPP: Для локального тестування серверної частини.

HeidiSQL: Для зручного управління базами даних.

Плагіни: JPA Buddy, Thymeleaf Plugin, IntelliJ Lombok Plugin для спрощення роботи з ORM, шаблонами та Lombok.

## ДОДАТОК В - ФРАГМЕНТ ПРОГРАМНОГО КОДУ

### // Java-контролер для реєстрації користувача

```
package kursovabd.controller;

import kursovabd.Repository.UserRepository;
import kursovabd.model.Users;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

@Controller
public class RegController {

    private final UserRepository userRepository;

    @Autowired
    public RegController(UserRepository userRepository) {
        this.userRepository = userRepository;
    }

    @GetMapping("/register")
    public String showRegistrationForm(Model model) {
        model.addAttribute("user", new Users());
        return "register";
    }

    @PostMapping("/register")
    public String register(@ModelAttribute Users user, BindingResult result,
Model model) {
        if (result.hasErrors()) {
            return "register";
        }

        userRepository.save(user);
        return "redirect:/login";
    }
}
```

### // Java-контролер для логіну користувача

```
package kursovabd.controller;

import kursovabd.Repository.UserRepository;
import kursovabd.model.Users;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

@Controller
public class LoginController {

    @Autowired
```



```

private UserRepository userRepository;

@GetMapping("/login")
public String showLoginForm(Model model) {
    model.addAttribute("user", new Users());
    return "login";
}

@PostMapping("/login")
public String login(@ModelAttribute Users user, Model model) {
    // Найти пользователя по логину
    Users foundUser = userRepository.findByLogin(user.getLogin());

    // Проверка на null
    if (foundUser != null) {
        // Проверка для администратора
        if ("admin".equals(user.getLogin()) &&
foundUser.getPassword().equals(user.getPassword())) {
            return "redirect:/admin";
        }
        // Проверка для других пользователей
        if (foundUser.getPassword().equals(user.getPassword())) {
            return "redirect:/Users/" + foundUser.getId() +
"/cabinet";
        }
    }

    // Если данные не прошли проверку
    model.addAttribute("user", user);
    model.addAttribute("error", "Invalid email or password");
    return "login";
}
}

```

**// Thymeleaf-шаблон для форми реєстрації**

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Registration</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            text-align: center;
            margin-top: 50px;
        }
        h1 {
            color: #333;
        }
        form {
            background-color: #fff;
            border-radius: 8px;
            padding: 20px;
            width: 300px;
            margin: 0 auto;
            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
        }
    </style>

```

```

        label {
            display: block;
            margin-bottom: 10px;
            color: #333;
        }
        input[type="text"],
        input[type="email"],
        input[type="password"] {
            width: calc(100% - 22px);
            padding: 10px;
            border-radius: 4px;
            border: 1px solid #ccc;
            margin-bottom: 0px;
        }
        input[type="submit"] {
            background-color: #007bff;
            color: #fff;
            padding: 10px 20px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }
        input[type="submit"]:hover {
            background-color: #0056b3;
        }
    </style>
</head>
<body>

<h1>Реєстрація</h1>

<form method="post" modelAttribute="user">

    <label for="firstName">Ім'я:</label>
    <input type="text" id="firstName" name="firstName"><br><br>

    <label for="secondName">Прізвище:</label>
    <input type="text" id="secondName" name="secondName"><br><br>

    <label for="email">Пошта:</label>
    <input type="email" id="email" name="email" required><br><br>

    <label for="address">Адреса:</label>
    <input type="text" id="address" name="address"><br><br>

    <label for="phoneNumber">Номер телефону:</label>
    <input type="text" id="phoneNumber" name="phoneNumber"><br><br>

    <label for="login">Логін:</label>
    <input type="text" id="login" name="login"><br><br>

    <label for="password">Пароль:</label>
    <input type="password" id="password" name="password"
required><br><br>

```

```

        <input type="submit" value="Реєстрація">
    </form>
    <p>Маєте аккаунт?</p>
    <p><a href="login">Логін</a></p>
    <p><a href="index.html">Головна сторінка</a></p>

</body>
</html>

```

## // Thymeleaf-шаблон для форми авторизації

```

<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>Login</title>
    <!-- Подключение стилей для оформления -->
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f8f9fa;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }

        .container {
            background-color: #fff;
            padding: 20px;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            text-align: center;
        }

        h1 {
            margin-bottom: 20px;
            color: #333;
        }

        label {
            font-weight: bold;
            margin-right: 10px;
        }

        input[type="text"],
        input[type="password"] {
            padding: 8px;
            margin-bottom: 15px;
            width: 250px;
            border-radius: 5px;
            border: 1px solid #ccc;
        }
    </style>

```

```

        input[type="submit"] {
            padding: 10px 20px;
            background-color: #007bff;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            transition: background-color 0.3s;
        }

        input[type="submit"]:hover {
            background-color: #0056b3;
        }

        p.error {
            color: #ff0000;
            margin-top: 10px;
        }

        a {
            color: #007bff;
            text-decoration: none;
        }

        a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>
<div class="container">
    <h1>Вхід</h1>
    <form method="post" th:action="@{/login}" th:object="${user}">
        <label for="login">Логін:</label>
        <input type="text" id="login" name="login" th:field="*{login}"
required><br><br>
        <label for="password">Пароль:</label>
        <input type="password" id="password" name="password"
th:field="*{password}" required><br><br>
        <input type="submit" value="Вхід">
    </form>

    <!-- Display error message if present -->
    <p th:if="${error}" class="error" th:text="${error}"></p>

    <p><a href="#">Головна сторінка</a></p>
</div>
</body>
</html>

```

## // Java-контролер відображення тарифів

```
package kursovabd.controller;

import kursovabd.model.Tariff_plans;
import kursovabd.Repository.TariffPlanRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import java.util.Optional;

@Controller
public class TariffPlanController {

    @Autowired
    private TariffPlanRepository tariffPlanRepository;

    @GetMapping("/Tariff_Plans")
    public String showTariffPlans(Model model) {
        Iterable<Tariff_plans> tariffPlans =
tariffPlanRepository.findAll();
        model.addAttribute("tariffPlans", tariffPlans);
        return "tariff_plans";
    }

    @DeleteMapping("/Tariff_Plans/delete/{id}")
    public String deleteTariffPlan(@PathVariable("id") Long id) {
        tariffPlanRepository.deleteById(id);
        return "redirect:/Tariff_Plans";
    }

    @GetMapping("/tariff_Plans/edit/{id}")
    public String showEditForm(@PathVariable("id") Long id, Model
model) {
        Tariff_plans tariffPlan = tariffPlanRepository.findById(id)
            .orElseThrow(() -> new
IllegalArgumentException("Invalid tariff plan ID: " + id));
        model.addAttribute("tariffPlan", tariffPlan);
        return "editTariffPlan";
    }

    @PostMapping("/tariff_Plans/update/{id}")
    public String updateTariffPlan(@PathVariable("id") Long id,
@ModelAttribute("tariffPlan") Tariff_plans updatedTariffPlan,
RedirectAttributes redirectAttributes) {
        Optional<Tariff_plans> existingOptional =
tariffPlanRepository.findById(id);
        if (!existingOptional.isPresent()) {
            throw new IllegalArgumentException("Invalid tariff plan
ID: " + id);
        }
        Tariff_plans existingTariffPlan = existingOptional.get();

        existingTariffPlan.setName(updatedTariffPlan.getName());
    }
}
```

```

        existingTariffPlan.setCost(updatedTariffPlan.getCost());
        existingTariffPlan.setSpeed(updatedTariffPlan.getSpeed());

        tariffPlanRepository.save(existingTariffPlan);
        redirectAttributes.addFlashAttribute("message", "Tariff plan
updated successfully!");

        return "redirect:/Tariff_Plans";
    }
    @PostMapping("/tariff_Plans/add")
    public String addTariffPlan(
        @RequestParam(value = "name",
defaultValue = "") String name,
        @RequestParam(value = "cost",
defaultValue = "0") double cost,
        @RequestParam(value = "speed",
defaultValue = "0") int speed,
        RedirectAttributes redirectAttributes)
    {
        Tariff_plans newTariffPlan = new Tariff_plans();

        newTariffPlan.setName(name);
        newTariffPlan.setCost(cost);
        newTariffPlan.setSpeed(speed);

        tariffPlanRepository.save(newTariffPlan);

        redirectAttributes.addFlashAttribute("message", "Tariff plan
added successfully!");
        return "redirect:/Tariff_Plans";
    }
}

```

**// Thymeleaf-шаблон для відображення тарифів**

```

<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet" type="text/css" href="/style.css">
    <meta charset="UTF-8">
    <title>Tariff Plans</title>
</head>
<body>
<h1>Тарифні плани</h1>
<table>
    <thead>
        <tr>
            <th>ID</th>
            <th>Назва</th>
            <th>Ціна</th>
            <th>Швидкість</th>
            <th>Дії</th>
        </tr>
    </thead>

```

```

<tbody>
<tr th:each="tariffPlans : ${tariffPlans}">
  <td th:text="${tariffPlans.id}"></td>
  <td th:text="${tariffPlans.name}"></td>
  <td th:text="${tariffPlans.cost}"></td>
  <td th:text="${tariffPlans.speed}"></td>
  <td>
    <a
th:href="@{/tariff_Plans/edit/{id} (id=${tariffPlans.id})}"><button>Редагувати</button></a>
    <button th:onclick="'deleteTariffPlans(' +
${tariffPlans.id} + ')'">Видалити</button>
  </td>
</tr>
</tbody>
</table>

<h2>Додати новий тарифний план</h2>
<form th:action="@{/tariff_Plans/add}" method="post">

  <label for="name">Назва:</label>
  <input type="text" id="name" name="name"><br><br>
  <label for="cost">Ціна:</label>
  <input type="number" id="cost" name="cost" step="0.01"><br><br>
  <label for="speed">Швидкість:</label>
  <input type="number" id="speed" name="speed"><br><br>
  <button type="submit">Додати</button>
</form>

<script>
function deleteTariffPlans(tariffPlansId) {
  fetch(`/Tariff_Plans/delete/${tariffPlansId}`, {
    method: 'DELETE'
  }).then(response => {
    if (response.ok) {
      return response.json(); // Parse response JSON
    } else {
      console.error('Failed to delete tariff plan');
    }
  }).then(data => {
    window.location.reload(); // Reload page after successful
deletion
  }).catch(error => {
    console.error('Error deleting tariff plan:', error);
  });
}
</script>
</body>
</html>

```