МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

Навчально-науковий інститут електричної інженерії
та інформаційних технологій

КАФЕДРА АВТОМАТИЗАЦІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

ЗВІТ

З ЛАБОРАТОРНИХ РОБІТ

З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

«Frontend-розробка»

Виконав студент групи КН-23-1

Полинько Ігор Миколайович

Перевірив старший викладач кафедри АІС Притчин О. С.

КРЕМЕНЧУК 2025

Лабораторна робота № 6

Тема: Робота з вебзапитами та API в JavaScript та React

Мета: Познайомитись і отримати головні навички роботи зі зберіганням даних в різних сховищах веббраузера в JavaScript та React

Виконання завдання лабораторної роботи:

Створіть проєкт React. Додаток має містити сторінку де виводитиметься інформація та поле введення.

Зробіть так, щоб при відкритті сторінки виводилась інформація з певного кукі. При введенні тексту в поле введення він має записуватись в той же кукі з якого читається інформація при відкритті. Якщо кукі немає – створіть його при введенні першого символа в поле введення. При відкритті виведіть, що інформації поки немає.

Переведіть проєкт на Local Storage, потім на Session Storage. Зверніть увагу на відмінності між усіма цими підходами. Для Session Storage додайте функціонал читання і виведення даних одразу після їх збереження, щоб було видно різницю між збереженими даними та даними при повторному відкритті сторінки.

Cookie:

```
import React, { useState, useEffect } from "react";
import { useCookies } from "react-cookie";

function App() {
  const [text, setText] = useState("");
  const [cookies, setCookie] = useCookies(["myText"]);

  useEffect(() => {
    if (cookies.myText) {
      setText(cookies.myText);
    } else {
      setText("Інформації поки немає");
    }
  }, [cookies]);

  const handleChange = (e) => {
    setText(e.target.value);
    setCookie("myText", e.target.value, { path: "/" });
  };
```

```
  return (
    <div style={{ display: "flex", flexDirection: "column", alignItems: "center",
marginTop: "50px" }}>
      <h1>Збереження даних у cookie</h1>
      <input
        type="text"
        value={text === "Інформації поки немає" ? "" : text}
        onChange={handleChange}
        placeholder="Введіть текст"
      />
      <p>{text}</p>
    </div>
  );
}

export default App;
```

Local Storage:

```
import React, { useState, useEffect } from "react";

function App() {
  const [text, setText] = useState("");

  useEffect(() => {
    const stored = localStorage.getItem("myText");
    if (stored) {
      setText(stored);
    } else {
      setText("Інформації поки немає");
    }
  }, []);

  const handleChange = (e) => {
    setText(e.target.value);
    localStorage.setItem("myText", e.target.value);
  };

  return (
    <div style={{ display: "flex", flexDirection: "column", alignItems: "center",
marginTop: "50px" }}>
      <h1>Збереження даних у Local Storage</h1>
      <input
        type="text"
        value={text === "Інформації поки немає" ? "" : text}
        onChange={handleChange}
        placeholder="Введіть текст"
      />
      <p>{text}</p>
    </div>
```

```
  );
}

export default App;
```

Session Storage:

```
import React, { useState, useEffect } from "react";

function App() {
  const [text, setText] = useState("");

  useEffect(() => {
    const stored = sessionStorage.getItem("myText");
    if (stored) {
      setText(stored);
    } else {
      setText("Інформації поки немає");
    }
  }, []);

  const handleChange = (e) => {
    setText(e.target.value);
    sessionStorage.setItem("myText", e.target.value);
  };

  return (
    <div style={{ display: "flex", flexDirection: "column", alignItems: "center",
marginTop: "50px" }}>
      <h1>Збереження даних у Session Storage</h1>
      <input
        type="text"
        value={text === "Інформації поки немає" ? "" : text}
        onChange={handleChange}
        placeholder="Введіть текст"
      />
      <p>{text}</p>
      <p>Дані зберігаються тільки до закриття вкладки</p>
    </div>
  );
}

export default App;
```

Результат:

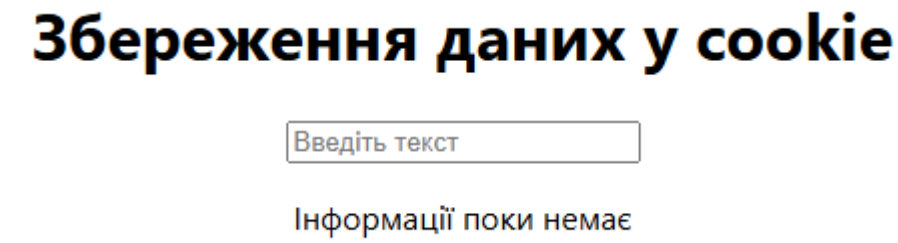

Рисунок 6.1 – Старт сторінки

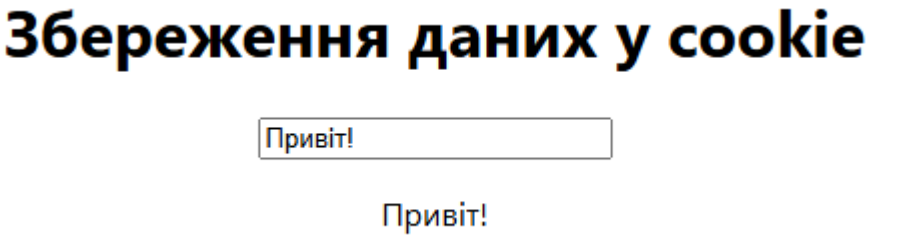

Рисунок 6.2 – Створення кукі

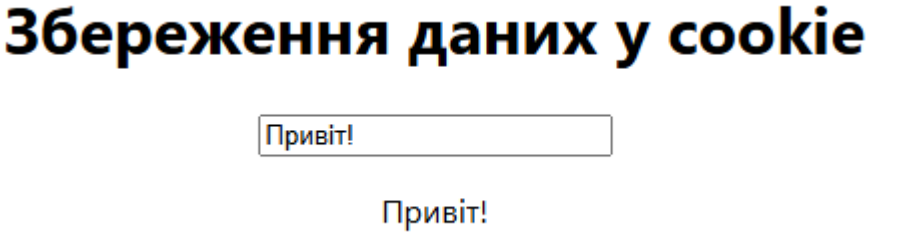

Рисунок 6.3 – Перевірка роботи cookie

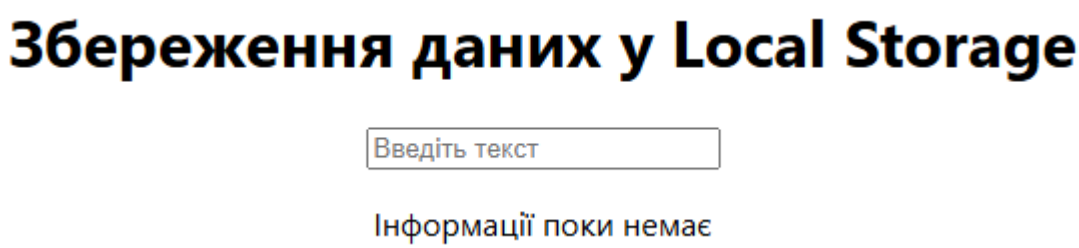

Рисунок 6.4 – Старт сторінки Local Storage

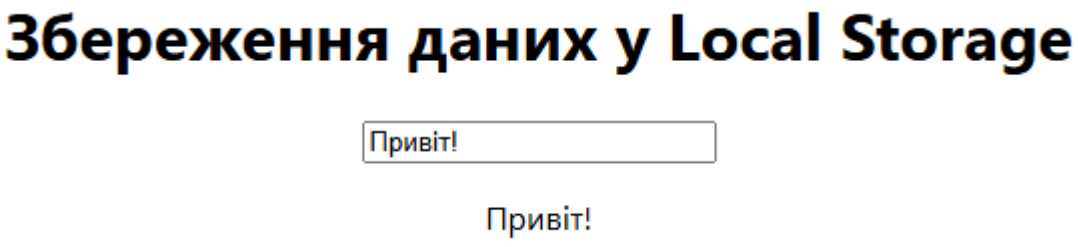

Рисунок 6.5 – Створення кукі Local Storage

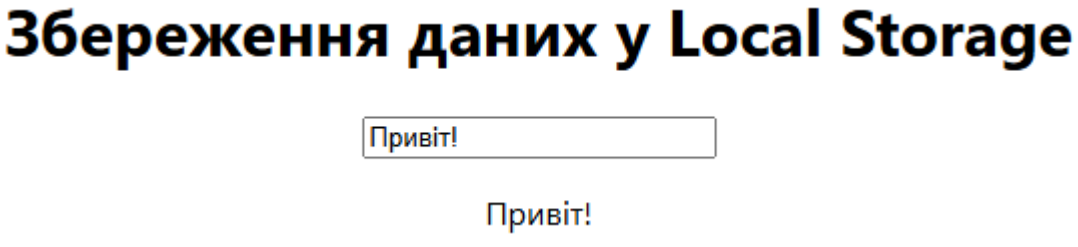

Рисунок 6.6 – Перевірка роботи Local Storage

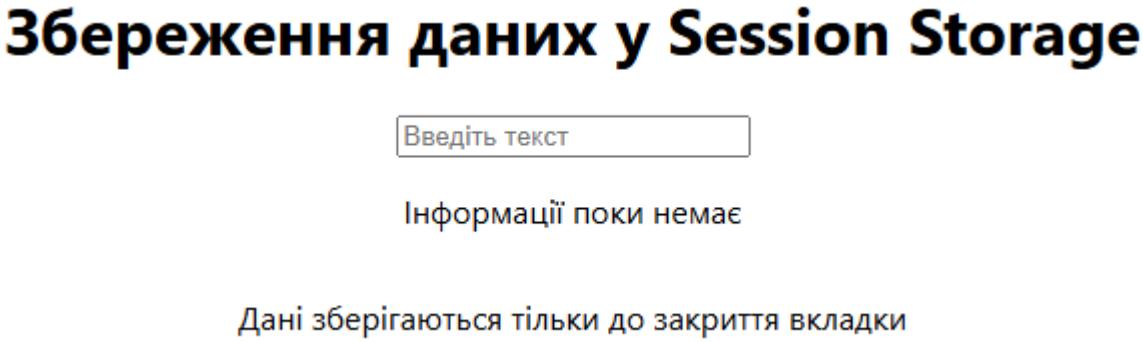

Рисунок 6.7 – Старт сторінки Session Storage

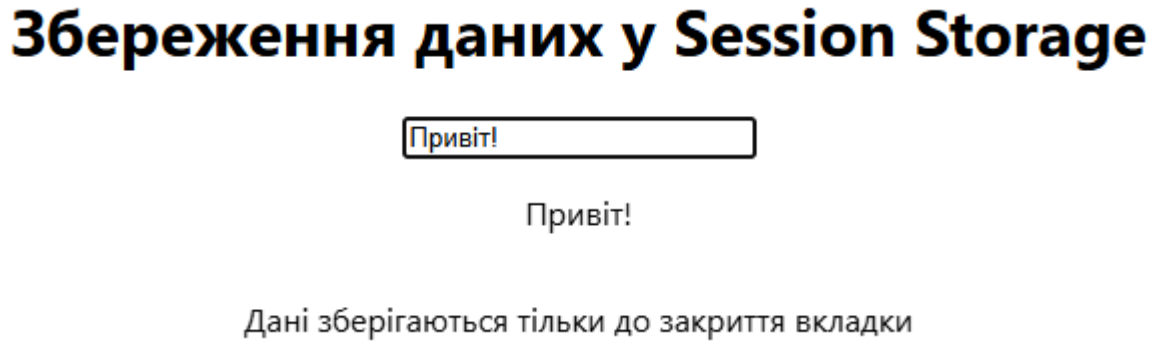

Рисунок 6.5 – Створення кукі Session Storage

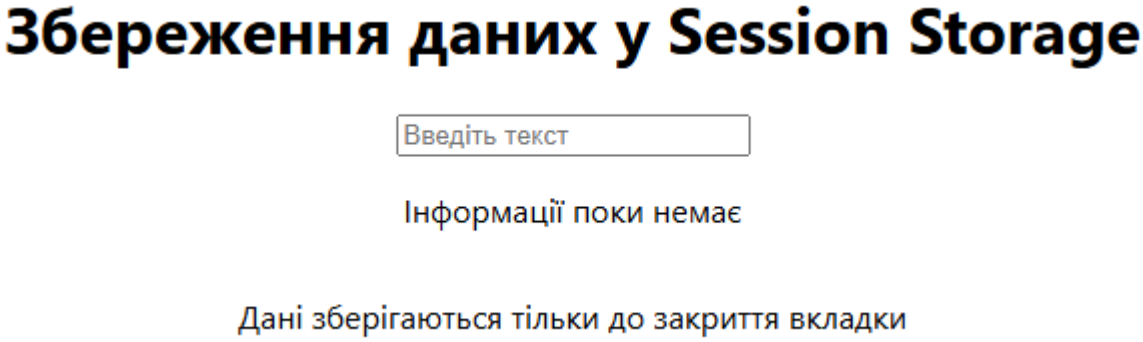

Рисунок 6.6 – Перевірка роботи Session Storage

Висновок: на цій лабораторній роботі ми познайомились і отримали головні навички роботи зі зберіганням даних в різних сховищах веббраузера в JavaScript та React.

Контрольні питання:

1. Що таке cookie?

Cookie – це невеликі текстові файли, що зберігаються на клієнтському пристрої браузером або сервером.

Вони використовуються для:

– автентифікації користувача,

– збереження сесій,

– відстеження дій користувача.

Характерні особливості:

– обмеження розміру (до ~4 КБ),

– автоматично відправляються з кожним HTTP-запитом до сервера.

2. Різниця між js-cookie та react-cookie

Бібліотека	Підхід	Особливості
js-cookie	Vanilla JS	Проста робота з cookie через методи set, get, remove. Не інтегрована з React-хуками.
react-cookie	React	Використовує хук useCookies, автоматично оновлює стан компонента при зміні cookie. Зручна для інтеграції з React-компонентами.

3. Що таке localStorage?

localStorage – це вбудований об’єкт браузера для зберігання даних у форматі ключ-значення.

Особливості:

– дані зберігаються постійно, навіть після закриття браузера,

– підтримує тільки рядки; для збереження об’єктів чи масивів потрібно використовувати JSON.stringify() і JSON.parse().

4. Основні методи для роботи з localStorage

– localStorage.setItem('ключ', 'значення') – зберігає пару ключ-значення.

– localStorage.getItem('ключ') – отримує значення за ключем.

– localStorage.removeItem('ключ') – видаляє пару ключ-значення.

– localStorage.clear() – очищає всі дані для поточного домену.

5. Відмінність між localStorage та sessionStorage

Параметр	**localStorage**	**sessionStorage**
Час зберігання	Постійно (після закриття браузера)	До закриття вкладки/вікна
Доступ	Через JS	Через JS
Використання	Для налаштувань, токенів, кешу	Для тимчасових даних під час сесії

6. Інтеграція localStorage та sessionStorage у React компоненти

– Використовуються хуки useState та useEffect.

– При завантаженні сторінки через useEffect дані читаються зі сховища і встановлюються у стан компонента (useState).

– При зміні даних у компоненті (onChange) виконується запис у localStorage або sessionStorage.

Приклад:

```
const [text, setText] = useState("");

useEffect(() => {
  const stored = localStorage.getItem("myText");
  if (stored) setText(stored);
}, []);

const handleChange = (e) => {
  setText(e.target.value);
  localStorage.setItem("myText", e.target.value);
};
```