Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет електроніки і комп’ютерних технологій

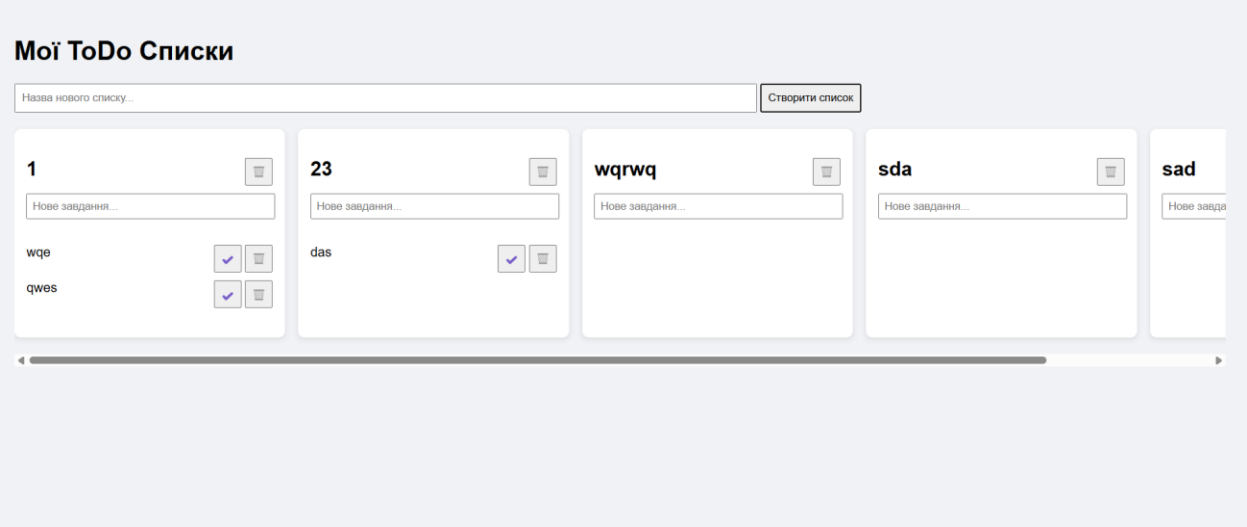
Лабораторна робота №6
З курсу “Веб програмування на стороні клієнта”
“ Порівняння розробки ToDo List за допомогою чистого JavaScript та React”

Виконав:
студент ФЕІ-25
Кайда Матвій
Перевірив:
доц. Анохін В. М.

Львів 2025

Мета: Дослідити відмінності між імперативним та декларативним підходами у розробці інтерфейсів. Ознайомитися з основними принципами React. Реалізувати ToDo List двома способами: За допомогою чистого JavaScript та за допомогою React Порівняти складність коду, гнучкість та розширюваність.

Common js:



Index.html:

```html
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>ToDo Lists</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="main-container">
    <h1>Мої ToDo Списки</h1>
    <input type="text" id="list-name-input" placeholder="Назва нового списку...">
    <button id="add-list">Створити список</button>
    <div id="lists-container"></div>
  </div>
  <script src="script.js"></script>
</body>
</html>
```

Script.js:

```
const listNameInput = document.getElementById('list-name-input');
const addListBtn = document.getElementById('add-list');
const listsContainer = document.getElementById('lists-container');

let todoLists = JSON.parse(localStorage.getItem('todoLists')) || [];

function saveAll() {
  localStorage.setItem('todoLists', JSON.stringify(todoLists));
}

function renderAllLists() {
  listsContainer.innerHTML = '';
  todoLists.forEach((list, listIndex) => {
    const listDiv = document.createElement('div');
    listDiv.className = 'todo-list';

    listDiv.innerHTML = `
      <h2>
        ${list.name}
        <button onclick="deleteList(${listIndex})">🗑</button>
      </h2>
      <input type="text" placeholder="Нове завдання..." onkeypress="handleAddTask(event, ${listIndex})">
      <ul class="task-list" id="task-list-${listIndex}"></ul>
    `;

    listsContainer.appendChild(listDiv);
    renderTasks(list.tasks, listIndex);
  });
}

function renderTasks(tasks, listIndex) {
  const taskListEl = document.getElementById(`task-list-${listIndex}`);
  taskListEl.innerHTML = '';

  tasks.forEach((task, taskIndex) => {
    const li = document.createElement('li');
    li.className = task.completed ? 'completed' : '';
    li.innerHTML = `
      <span>${task.text}</span>
      <div>
        <button onclick="toggleComplete(${listIndex}, ${taskIndex})">✔</button>
        <button onclick="deleteTask(${listIndex}, ${taskIndex})">🗑</button>
      </div>
    `;
    taskListEl.appendChild(li);
```

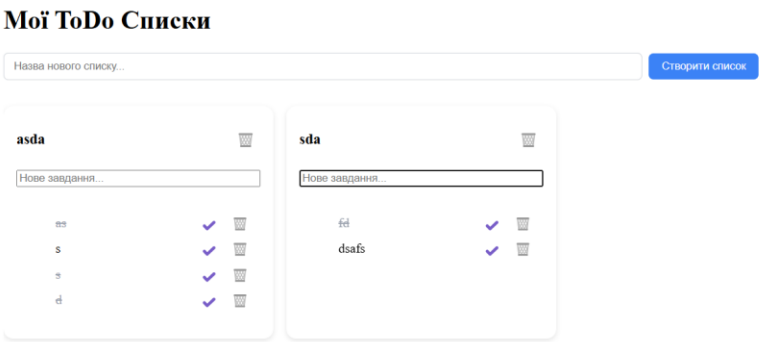
```
  });
}

function addList() {
  const name = listNameInput.value.trim();
  if (name) {
    todoLists.push({ name, tasks: [] });
    listNameInput.value = '';
    saveAll();
    renderAllLists();
  }
}

function handleAddTask(event, listIndex) {
  if (event.key === 'Enter') {
    const input = event.target;
    const text = input.value.trim();
    if (text) {
      todoLists[listIndex].tasks.push({ text, completed: false });
      input.value = '';
      saveAll();
      renderAllLists();
    }
  }
}

function toggleComplete(listIndex, taskIndex) {
  const task = todoLists[listIndex].tasks[taskIndex];
  task.completed = !task.completed;
  saveAll();
  renderAllLists();
}

function deleteTask(listIndex, taskIndex) {
  todoLists[listIndex].tasks.splice(taskIndex, 1);
  saveAll();
  renderAllLists();
}

function deleteList(listIndex) {
    todoLists.splice(listIndex, 1);
    saveAll();
    renderAllLists();
}

addListBtn.addEventListener('click', addList);
renderAllLists();
```

Styles.css:

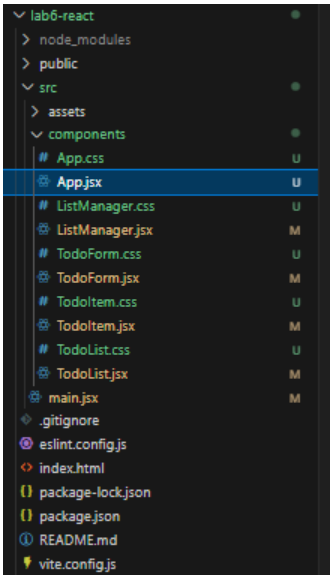
```css
body {
    font-family: Arial, sans-serif;
    background: #f0f2f5;
    padding: 30px;
    margin: 0;
}

.main-container {
    max-width: 100%;
    overflow-x: hidden;
}

#list-name-input {
    width: 60%;
    padding: 8px;
}

#add-list {
    padding: 8px;
}

#lists-container {
    display: flex;
    overflow-x: auto;
    gap: 16px;
    padding: 20px 0;
}

.todo-list {
    flex: 0 0 25%;
    min-width: 250px;
    max-width: 300px;
    background: white;
    padding: 15px;
    border-radius: 8px;
    box-shadow: 0 2px 6px rgba(0,0,0,0.1);
    display: flex;
    flex-direction: column;
}

.todo-list h2 {
    display: flex;
    justify-content: space-between;
    margin-bottom: 10px;
}

.todo-list input {
    width: calc(100% - 12px);
    padding: 6px;
    margin-bottom: 10px;
}
```

```css
.todo-list button {
  padding: 6px;
}

.task-list {
  list-style: none;
  padding-left: 0;
  flex-grow: 1;
  overflow-y: auto;
}

.task-list li {
  display: flex;
  justify-content: space-between;
  padding: 5px 0;
}

.completed {
  text-decoration: line-through;
  color: gray;
}
```

React.js:



Структура проекту:


lab6-react
node_modules
public
src
assets
components
App.css U
App.jsx U
ListManager.css U
ListManager.jsx M
TodoForm.css U
TodoForm.jsx M
TodoItem.css U
TodoItem.jsx M
TodoList.css U
TodoList.jsx M
main.jsx M
.gitignore
eslint.config.js
index.html
package-lock.json
package.json
README.md
vite.config.js

App.jsx:

```jsx
import { useState, useEffect } from "react";
import ListManager from "./ListManager.jsx";
import TodoList from "./TodoList.jsx";

export default function App() {
  const [lists, setLists] = useState([]);

  useEffect(() => {
    const saved = localStorage.getItem("todoLists");
    if (saved) {
      setLists(JSON.parse(saved));
    }
  }, []);

  useEffect(() => {
    localStorage.setItem("todoLists", JSON.stringify(lists));
  }, [lists]);

  const addList = (name) => {
    setLists([...lists, { name, tasks: [] }]);
  };

  const deleteList = (index) => {
    const updated = [...lists];
    updated.splice(index, 1);
    setLists(updated);
  };

  const updateTasks = (index, tasks) => {
    const updated = [...lists];
    updated[index].tasks = tasks;
    setLists(updated);
  };

  return (
    <div className="app">
      <h1>Мої ToDo Списки</h1>
      <ListManager onAdd={addList} />
      <div className="lists-container">
        {lists.map((list, index) => (
          <TodoList
            key={index}
            name={list.name}
            tasks={list.tasks}
            onUpdate={(tasks) => updateTasks(index, tasks)}
            onDelete={() => deleteList(index)}
          />
        ))}
      </div>
    </div>
  );
}
```

App.css:

```css
.app {
    padding: 1rem;
  }

  h1 {
    font-size: 24px;
    font-weight: bold;
    margin-bottom: 1rem;
  }
```

ListManager.jsx:

```jsx
import { useState } from "react";
import './ListManager.css';

export default function ListManager({ onAdd }) {
  const [name, setName] = useState("");

  const handleAdd = () => {
    if (name.trim()) {
      onAdd(name);
      setName("");
    }
  };

  return (
    <div className="list-manager">
      <input
        className=""
        placeholder="Назва нового списку..."
        value={name}
        onChange={(e) => setName(e.target.value)}
      />
      <button onClick={handleAdd}>Створити список</button>
    </div>
  );
}
```

ListManager.css:

```css
.list-manager {
    display: flex;
    align-items: center;
    gap: 0.5rem;
    margin-bottom: 1rem;
  }

  .list-manager input {
    padding: 0.5rem 0.75rem;
    border: 1px solid #d1d5db;
    border-radius: 0.375rem;
    width: 50%;
  }

  .list-manager button {
    background-color: #3b82f6;
    color: white;
    padding: 0.5rem 1rem;
    border: none;
    border-radius: 0.375rem;
    cursor: pointer;
    transition: background-color 0.2s;
  }

  .list-manager button:hover {
    background-color: #2563eb;
  }
```

TodoForm.jsx:

```jsx
import { useState } from "react";
import './TodoForm.css';

export default function TodoForm({ onAdd }) {
  const [text, setText] = useState("");

  const handleKeyPress = (e) => {
    if (e.key === "Enter" && text.trim()) {
      onAdd(text);
      setText("");
    }
  };

  return (
    <input
      className="todo-form"
      placeholder="Нове завдання..."
      value={text}
      onChange={(e) => setText(e.target.value)}
      onKeyPress={handleKeyPress}
    />
  );
}
```

TodoForm.css:

```css
.todo-form {
  margin-bottom: 0.75rem;
}

.todo-form input {
  width: 100%;
  padding: 0.5rem 0.75rem;
  border: 1px solid #d1d5db;
  border-radius: 0.375rem;
}
```

TodoItem.jsx:

```jsx
import './TodoItem.css';
export default function TodoItem({ task, onToggle, onDelete }) {
  return (
    <li className={`todo-item ${task.completed ? "completed" : ""}`}>
      <span>{task.text}</span>
      <div className="actions">
        <button onClick={onToggle}>✔</button>
        <button onClick={onDelete}>🗑</button>
      </div>
    </li>
  );
}
```

TodoItem.css:

```css
.todo-item {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 0.25rem 0.5rem;
    border-radius: 0.375rem;
}

.todo-item.completed {
    text-decoration: line-through;
    color: #9ca3af;
}

.todo-item .actions button {
    background: none;
    border: none;
    font-size: 1rem;
    cursor: pointer;
    margin-left: 0.25rem;
}

.todo-item .actions button:first-child {
    color: #16a34a;
}

.todo-item .actions button:first-child:hover {
    color: #15803d;
}

.todo-item .actions button:last-child {
    color: #ef4444;
}

.todo-item .actions button:last-child:hover {
    color: #b91c1c;
}
```

TodoList.jsx:

```jsx
import TodoForm from "./TodoForm.jsx";
import TodoItem from "./TodoItem.jsx";
import './TodoList.css';

export default function TodoList({ name, tasks, onUpdate, onDelete }) {
  const addTask = (text) => {
    onUpdate([...tasks, { text, completed: false }]);
  };

  const toggleTask = (index) => {
    const updated = [...tasks];
    updated[index].completed = !updated[index].completed;
    onUpdate(updated);
  };

  const deleteTask = (index) => {
    const updated = [...tasks];
    updated.splice(index, 1);
    onUpdate(updated);
  };

  return (
    <div className="todo-list">
      <div className="todo-list-header">
        <h2>{name}</h2>
        <button onClick={onDelete}>▒</button>
      </div>
      <TodoForm onAdd={addTask} />
      <ul className="flex-grow overflow-y-auto space-y-2">
        {tasks.map((task, index) => (
          <TodoItem
            key={index}
            task={task}
            onToggle={() => toggleTask(index)}
            onDelete={() => deleteTask(index)}
          />
        ))}
      </ul>
    </div>
  );
}
```

TodoList.css:

```css
.todo-list {
    background-color: white;
    padding: 1rem;
    border-radius: 0.75rem;
    box-shadow: 0 2px 6px rgba(0,0,0,0.1);
    width: 25%;
    min-width: 250px;
    max-width: 300px;
    display: flex;
    flex-direction: column;
  }

  .todo-list-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 0.75rem;
  }

  .todo-list-header h2 {
    font-size: 1.125rem;
    font-weight: 600;
  }

  .todo-list-header button {
    color: #ef4444;
    background: none;
    border: none;
    font-size: 1.1rem;
    cursor: pointer;
  }

  .todo-list-header button:hover {
    color: #b91c1c;
  }

  .lists-container {
    display: flex;
    gap: 1rem;
    padding-top: 1rem;
    padding-bottom: 0.5rem;
    overflow-x: auto;
  }
```

main.jsx:

```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './components/App.jsx'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

Висновок: Порівнюючи обидва проекти на common js та react.js можна побачити, що коду на common js вийшло значно менше, проте він не є так гарно структурований, як на react.js. В такому маленькому проекті не скалдно розібратись, тому структуризація не настільки важлива. Крім того, на common js треба повністю самому оптимізовувати проект, а react це вже робить, перерендрюючи компоненти тільки якщо вона змінюється.