

## JS I DOM NA PRZYKŁADZIE LISTY TODO

### SPIS TREŚCI

Spis treści .....	1
Cel zajęć.....	1
Rozpoczęcie.....	1
Uwaga .....	2
Wymagania.....	2
Strona HTML .....	2
Klasa Todo .....	3
Dodawanie pozycji listy .....	4
Usuwanie pozycji listy .....	5
Edycja pozycji listy.....	6
Odczyt / Zapis LocalStorage .....	8
Wyszukiwanie.....	9
Commit projektu do GIT.....	11
Podsumowanie.....	11

### CEL ZAJĘĆ

Celem głównym zajęć jest zdobycie następujących umiejętności:

- przemieszczania się po drzewie DOM;
- dodawania, usuwania, edytowania elementów drzewa DOM.

W praktycznym wymiarze utworzona zostanie dynamiczna lista czynności do zrobienia (lista To Do).

### ROZPOCZĘCIE

Rozpoczęcie zajęć. Powtórzenie metod przemieszczania się po drzewie DOM.

Wejściówka?

## UWAGA

Ten dokument aktywnie wykorzystuje niestandardowe właściwości. Podobnie jak w LAB A wejdź do Plik -> Informacje -> Właściwości -> Właściwości zaawansowane -> Niestandardowe i zaktualizuj pola. Następnie uruchom ten dokument ponownie lub Ctrl+A -> F9.

## WYMAGANIA

W ramach LAB B przygotowane powinny zostać:

- pojedyncza strona HTML ze skryptem ładowanym z zewnętrznego pliku JS
- lista zadań
- na dole listy pole tekstowe do dodawania nowych zadań, pole typu data/czas do określenia terminu wykonania zadania, przycisk dodawania zadania
- walidacja nowych zadań: co najmniej 3 znaki, nie więcej niż 255 znaków, data musi być pusta albo w przyszłości
- na górze listy pole wyszukiwarki
- po wpisaniu w wyszukiwarkę co najmniej 2 znaków na liście wyświetlają się wyłącznie pozycje zawierające wpisaną w wyszukiwarkę frazę
- wyszukiwana fraza zostaje wyróżniona w każdym wyniku wyszukiwania
- kliknięcie na dowolną pozycję listy zmienia ją w pole edycji; kliknięcie poza pozycję listy zapisuje zmiany
- obok każdej pozycji listy znajduje się przycisku Usuń / Śmiertnik
- wpisy na liście zapisują się do Local Storage
- po odświeżeniu strony lista wypełnia się wpisami z Local Storage

Mockupy:

chocolate	2000-01-01	X
macaroon		X
chupa chups		X
candy canes	2000-01-05	X
bon bons		X

macaroon	X
bon bons	X

## STRONA HTML

Prace rozpocznij od implementacji HTML z danymi wpisanymi „na sztywno”. Upewnij się, że wstawione zostały wszystkie wymagane elementy – pole wyszukiwarki, lista, pole dodawania, przycisk usuwania.

Wstaw zrzut ekranu przedstawiający stronę HTML z polem wyszukiwarki, listą, polem dodawania, przyciskami usuwania:

The screenshot shows a Windows desktop environment with a taskbar at the bottom. The taskbar includes icons for the Start button, File Explorer, Task View, and a system tray icon. The date and time are shown as 16:36 23.10.2023. The main window is titled 'Title' and displays a 'To Do' application. The interface includes a search bar labeled 'Search', a 'Tasks' section listing two items with delete buttons, and an 'Add Task' form with fields for 'Task' (containing 'trytrytrytryr'), 'dd.mm.yyyy' (containing '19.10.2023'), and a 'Add Task' button.

## KLASA TODO

Pierwszym instynktem może być chęć dodania zachowań bezpośrednio do elementów listy. Chociaż na krótką metę wydaje się być to najprostsze rozwiązanie, za chwilę okaże się krótkowzroczne i trudne do implementacji przy kolejnych punktach 😊

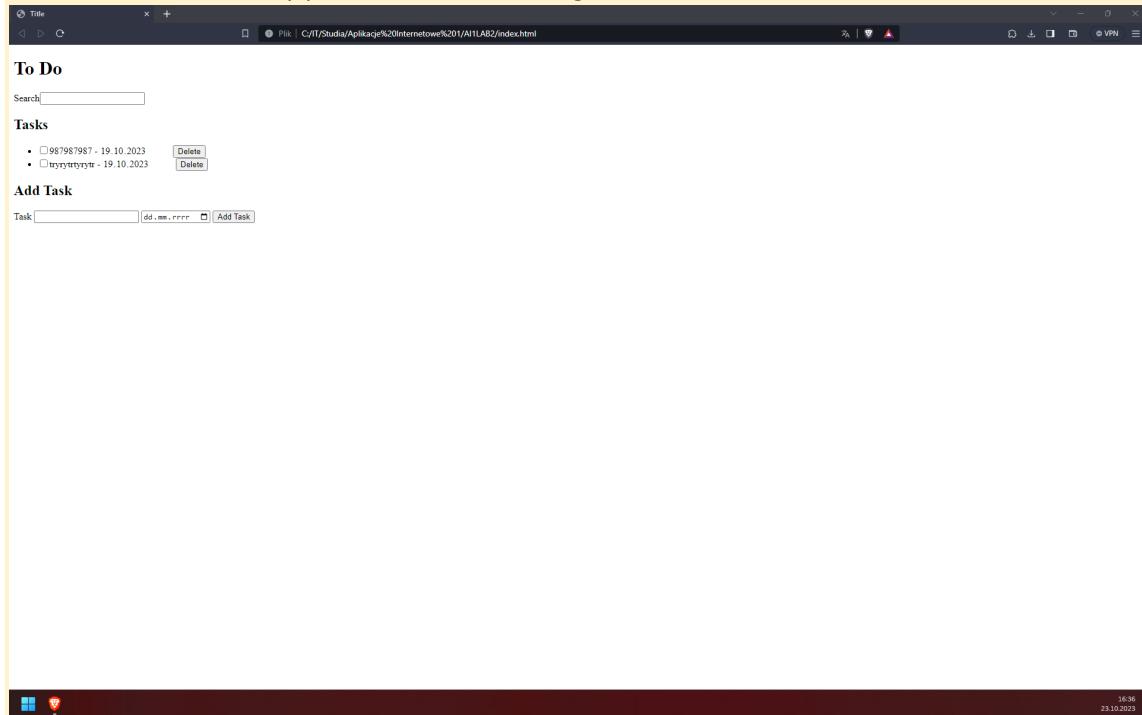
Najlepszym sposobem rozwiązania tego laboratorium jest utworzenie klasy Todo (albo po prostu obiektu z kilkoma metodami). Bez względu na przyjętą strategię, należy w tym nowoutworzonym bycie utworzyć tablicę `tasks` oraz metodę `draw()`, która wyczyszczy `div` z obecną wizualizacją zadań do zrobienia i wygeneruje ją na nowo na podstawie tablicy `tasks`.

W celu sprawdzenia poprawności działania, najlepiej dostać się do tablicy `tasks` i edytować jej zawartość, po czym ręcznie wywołać metodę `draw()`. Jeśli zawartość listy wyrenderuje się na nowo poprawnie – możemy iść dalej!

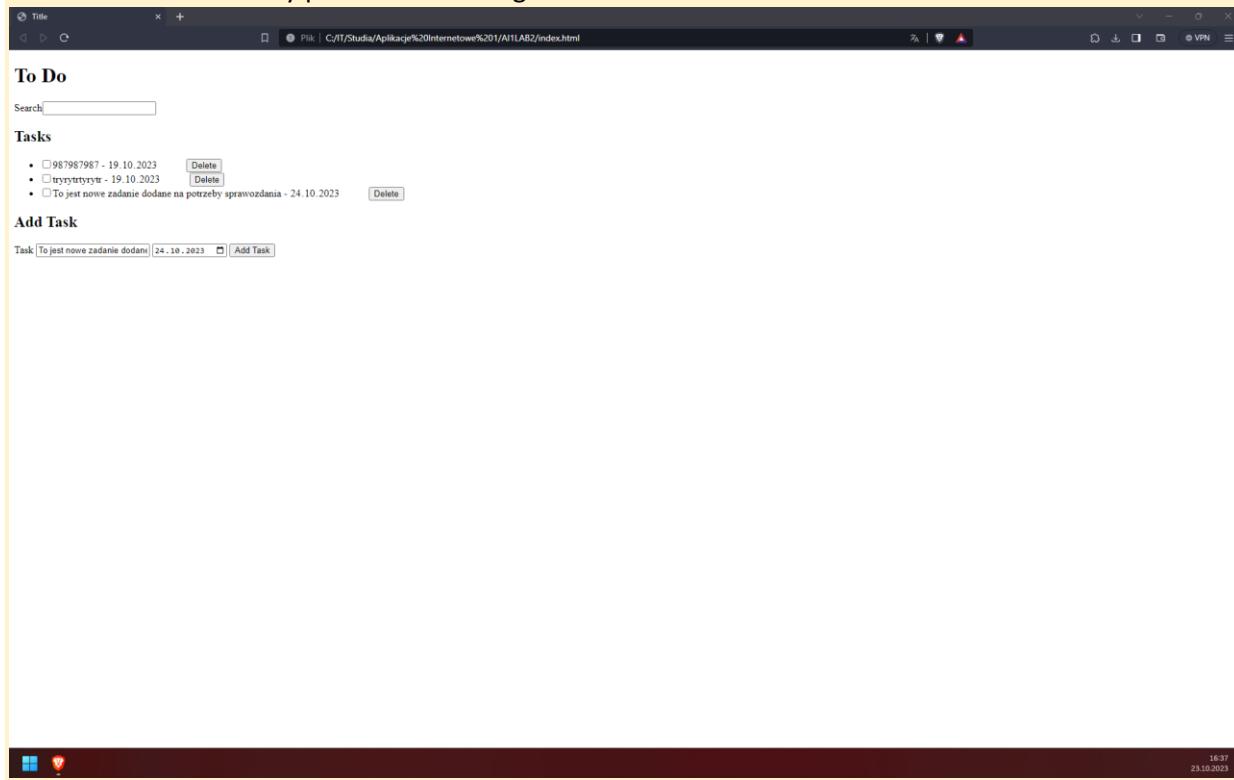
Zaimplementuj dodawanie, usuwanie, edycję pozycji listy – wszystko modyfikujące tablicę `tasks` i wywołujące na koniec metodę `draw()`.

## DODAWANIE POZYCJI LISTY

Wstaw zrzut ekranu listy przed dodaniem nowego zadania:



Wstaw zrzut ekranu listy po dodaniu nowego zadania:



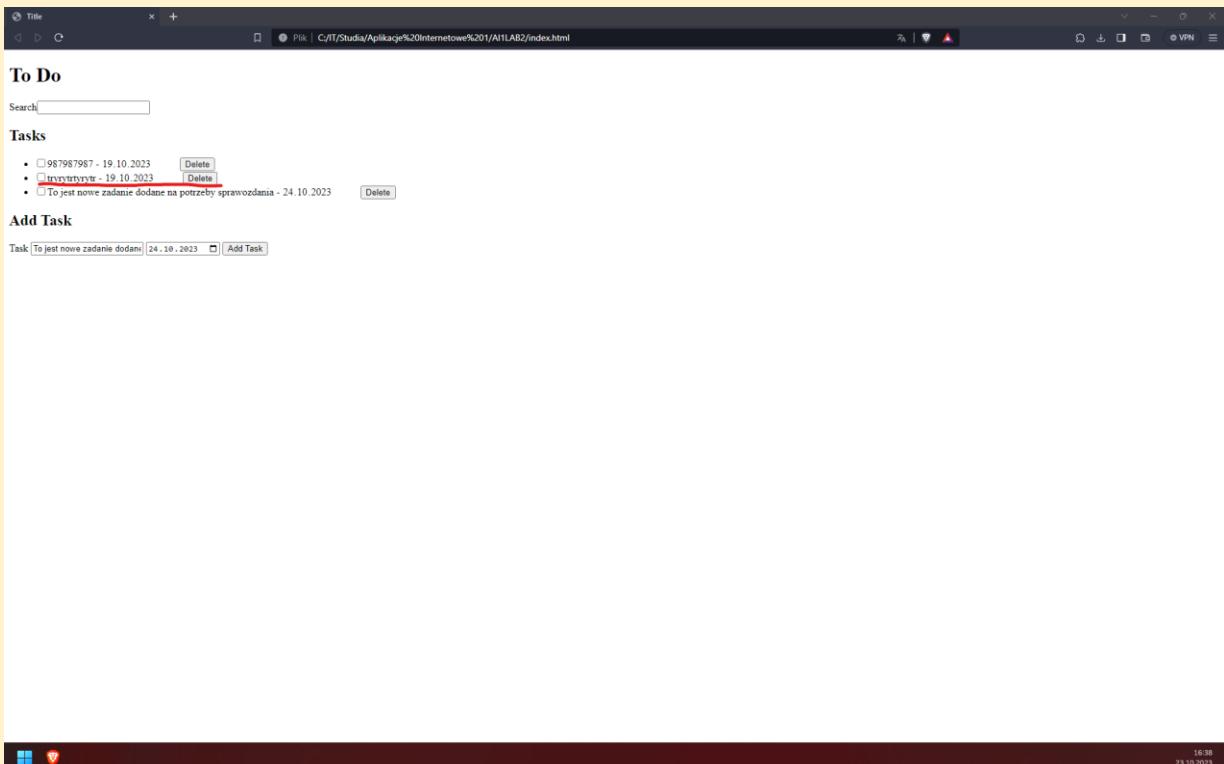
Punkty:

0

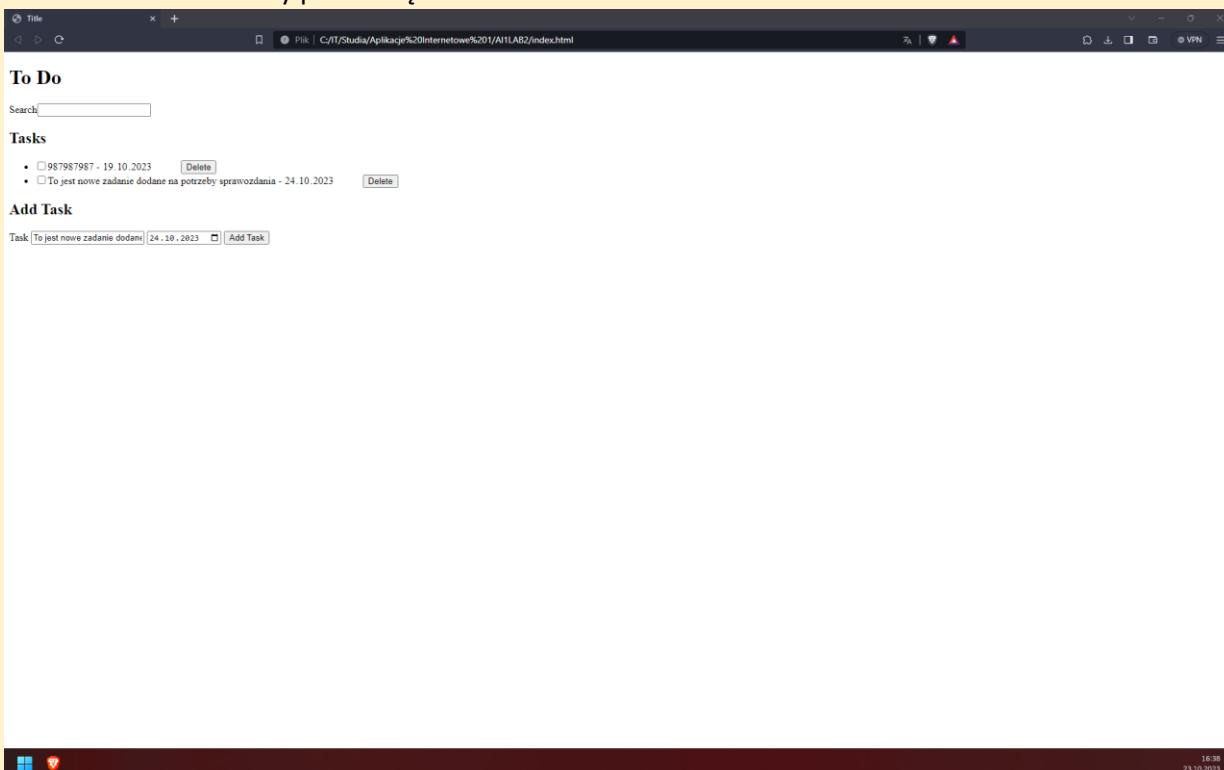
1

## USUWANIE POZYCJI LISTY

Wstaw zrzut ekranu listy przed usunięciem wybranego zadania:



Wstaw zrzut ekranu listy po usunięciu zadania:



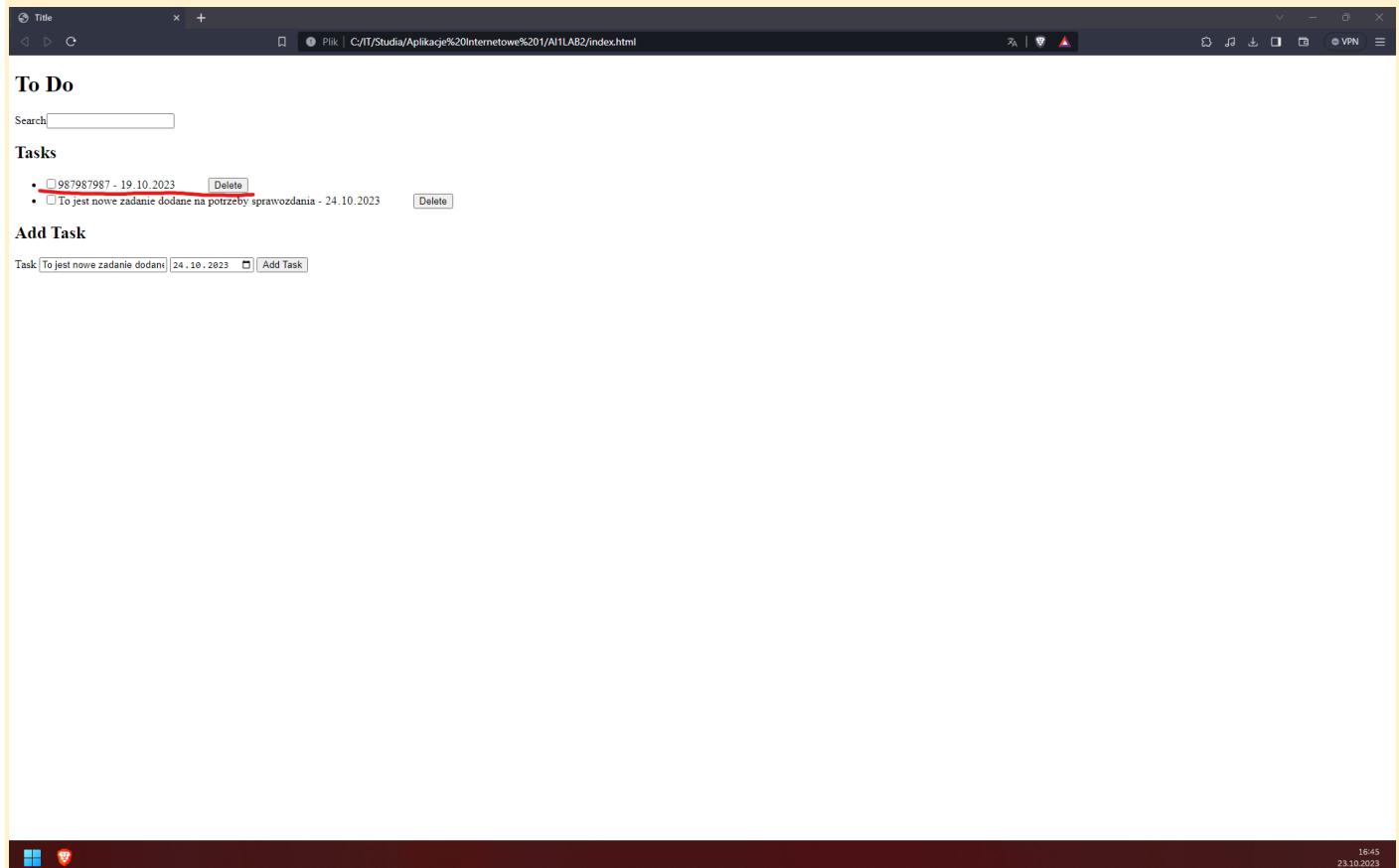
Punkty:

0

1

## EDYCJA POZYCJI LISTY

Wstaw zrzut ekranu listy przed edycją wybranego zadania:



Wstaw zrzut ekranu listy w trakcie edytowania zadania i daty:

The screenshot shows a browser window with the title bar 'Title'. The address bar shows the path 'Plik | C:/IT/Studia/Aplikacje%20Internetowe%20'. The main content area displays a 'To Do' list with two items:

- To jest edycja zadania - 28.10.2023 Edit Delete
- To jest nowe zadanie dodane na potrzeby sprawozdania - 24.10.2023 Delete

## To Do

Search

### Tasks

- To jest edycja zadania - 28.10.2023 Edit Delete
- To jest nowe zadanie dodane na potrzeby sprawozdania - 24.10.2023 Delete

### Add Task

Task   Add Task

Wstaw zrzut ekranu listy po edycji zadania i daty. Upewnij się, że dane się zapisały i zadanie jest zmienione:

The screenshot shows a browser window with the title bar 'Title'. The address bar shows the path 'Plik | C:/IT/Studia/Aplikacje%20Internetowe%201/AI1LAB2/index.html'. The main content area displays a 'To Do' list with the same two items as before, but the date has been changed to '24.10.2023'.

## To Do

Search

### Tasks

- To jest edycja zadania - 28.10.2023 Delete
- To jest nowe zadanie dodane na potrzeby sprawozdania - 24.10.2023 Delete

### Add Task

Task   Add Task

Punkty:

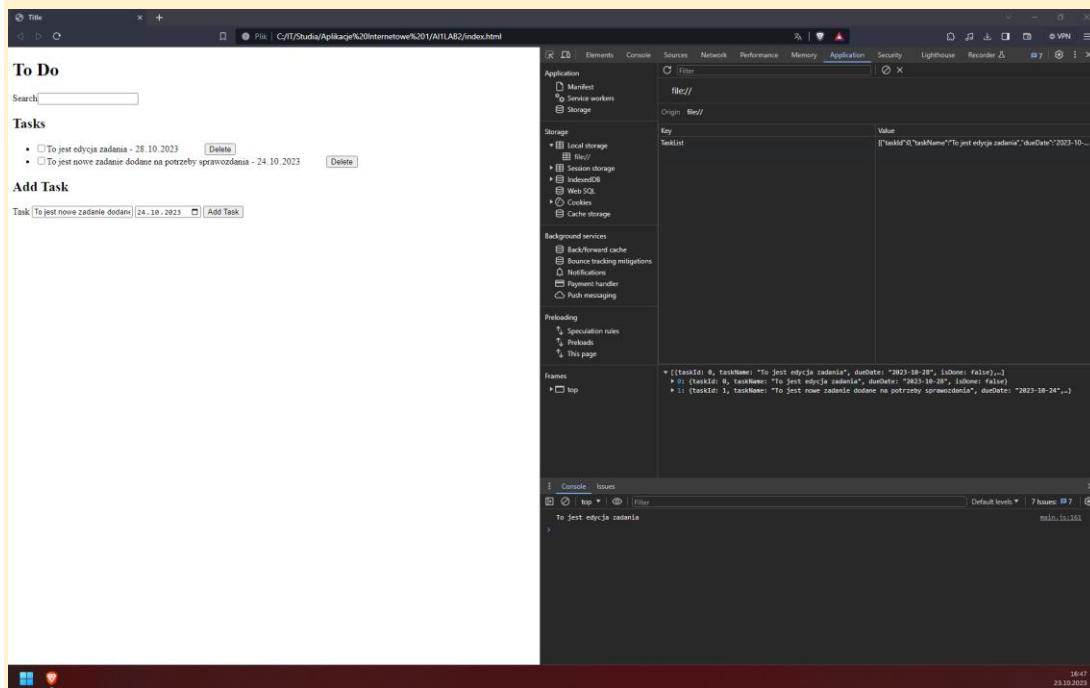
0

1

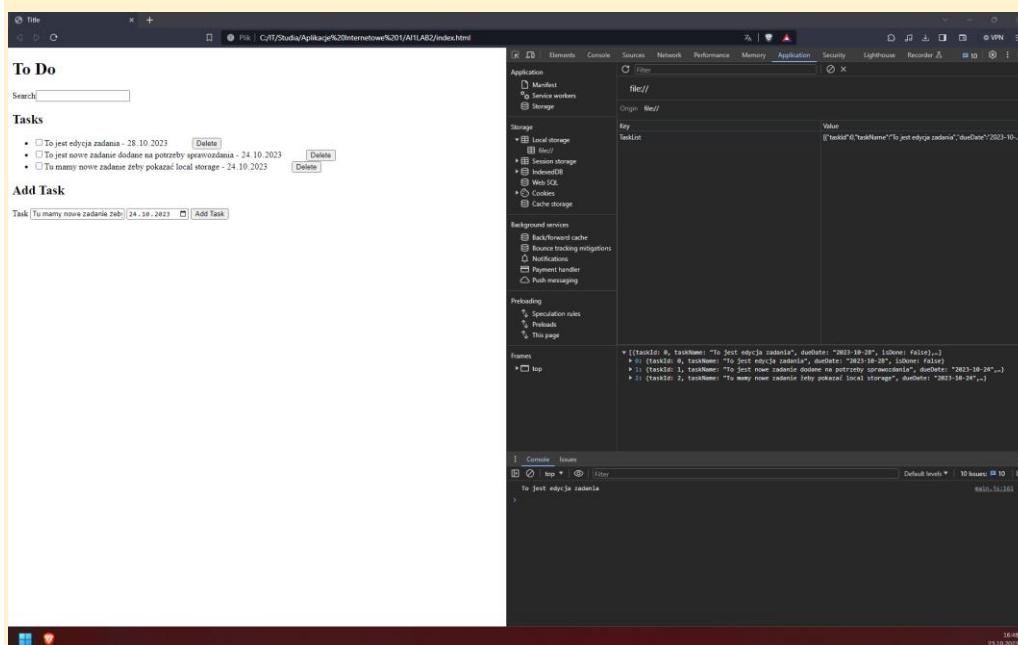
## ODCZYT / ZAPIS LOCALSTORAGE

Zastosowanie klasy Todo w realizacji tego laboratorium pozwala w bardzo łatwy sposób odczytywać i zapisywać stan listy do pamięci przeglądarki. Wystarczy serializacja / deserializacja za pomocą `JSON.parse()` i `JSON.stringify()`.

Wstaw zrzuty ekranu przedstawiające wygląd listy i zawartość local storage gdy na liście są pewne zadania:



Wstaw zrzuty ekranu przedstawiające wygląd listy i zawartość local storage po dodaniu nowej pozycji listy. Upewnij się, że widoczne w local storage są dane dotyczące nowego zadania:



Punkty:	0	1
---------	---	---

## WYSZUKIWANIE

Na koniec zostało filtrowanie wyników. Proponowanym podejściem do tego tematu jest umieszczenie w klasie Todo właściwości `term` – frazy wyszukiwanej przez użytkownika. Następnie można utworzyć metodę `getFilteredTasks`, albo getter `filteredTasks`, która zwracać będzie te elementy tablicy `tasks`, które odpowiadają zapytaniu. Można użyć funkcji wyższego rzędu `filter()`.

Wstaw zrzut ekranu listy, gdy pole wyszukiwania jest puste:

## To Do

Search

## Tasks

- To jest edycja zadania - 28.10.2023 Delete
- To jest nowe zadanie dodane na potrzeby sprawozdania - 24.10.2023 Delete
- Tu mamy nowe zadanie żeby pokazać local storage - 24.10.2023 Delete

## Add Task

Task  dd.mm.yyyy

Wstaw zrzut ekranu listy, gdy w polu wyszukiwania wpisano wystarczająco dużo znaków, by zadziałało filtrowanie. Upewnij się, że chociaż 2 wyniki będą wciąż widoczne:

**To Do**

Search To

**Tasks**

- To jest edycja zadania - 28.10.2023
- To jest nowe zadanie dodane na potrzeby sprawozdania - 24.10.2023

**Add Task**

Task  dd.mm.yyyy

Punkty:	0	1
---------	---	---

Wstaw zrzut ekranu przedstawiający podświetlenie szukanej frazy w wynikach wyszukiwania, przykładowo dla frazy imp i zadania implementacja otrzymujemy: implementacja:

**To Do**

Search zad

**Tasks**

- To jest edycja zadania - 28.10.2023
- To jest nowe zadanie dodane na potrzeby sprawozdania - 24.10.2023
- Tu mamy nowe zadanie żeby pokazać local storage - 24.10.2023

**Add Task**

Task  dd.mm.yyyy

Punkty:	0	1
---------	---	---

## COMMIT PROJEKTU DO GIT

Zacommituj i pushnij swoje rozwiązanie do repozytorium GIT.

Upewnij się, czy wszystko dobrze się wysłało. Jeśli tak, to z poziomu przeglądarki utwórz branch o nazwie `lab-b` na podstawie głównej gałęzi kodu.

Podaj link do brancha `lab-b` w swoim repozytorium:

...link, np <https://github.com/1Batrex1/ai1-lab1/tree/lab-b>

## PODSUMOWANIE

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.

Nauczyłem się jak używać local storage oraz jak za pomocą klas robić użyteczne programy w js

Zweryfikuj kompletność sprawozdania. Utwórz PDF i wyślij w terminie.