

Dokumentacja aplikacji "Notatnik"

Opis projektu

"Notatnik" to prosta aplikacja internetowa, która umożliwia użytkownikowi tworzenie, edytowanie, formatowanie oraz zarządzanie notatkami. Aplikacja została stworzona z wykorzystaniem technologii HTML, CSS oraz JavaScript.

Zawartość plików

Plik: `index.html`

The screenshot displays the 'Notatnik' web application interface. At the top, the title 'Notatnik™' is prominently displayed. Below it, a sorting section labeled 'Sortuj po:' includes three buttons: 'Tytule', 'Kategorii', and 'Przypiętych'. A formatting toolbar follows, featuring icons for bold (B), italic (I), underline (U), strikethrough (ABC), bullet points, and four color selection boxes (yellow, blue, red, black). Below the toolbar is a text input field with the placeholder 'Wpisz tytuł notatki'. Underneath this is a larger text area for the note's content. A category dropdown menu is set to 'Praca'. Two green buttons, 'Dodaj' and 'Usuń wszystko', are positioned below the category menu. The interface shows two example notes. The first note has the title 'tytul1', content 'tresc1', and category 'Praca'. The second note has the title 'tytul2', content 'tresc2', and category 'Szkoła'. Each note card includes a checkbox, a green checkmark icon, an edit icon (pencil), and a red 'X' icon for deletion.

Ten plik zawiera strukturę HTML aplikacji, definiując układ interfejsu użytkownika oraz elementy interaktywne.

Najważniejsze sekcje:

1. Nagłówek strony

- Tytuł aplikacji: "Notatnik™"

Notatnik™

2. Sortowanie notatek

- Pasek sortowania notatek według kryteriów: tytuł, kategoria, przypięte notatki.

Sortuj po:

Tytule

Kategorii

Przypiętych

3. Pasek narzędzi do formatowania tekstu

- Przyciski umożliwiające formatowanie tekstu: pogrubienie, kursywa, podkreślenie, przekreślenie, wypunktowanie oraz zmiana koloru tekstu.

B

/

U

S

•



4. Formularz notatki

- Pole tekstowe do wprowadzenia tytułu notatki.
- Główne pole edycyjne z możliwością formatowania treści.
- Lista rozwijana do wyboru kategorii notatki (z możliwością dodania własnej kategorii).

Wpisz tytuł notatki

Kategoria

Praca



5. Przyciski zarządzania notatkami

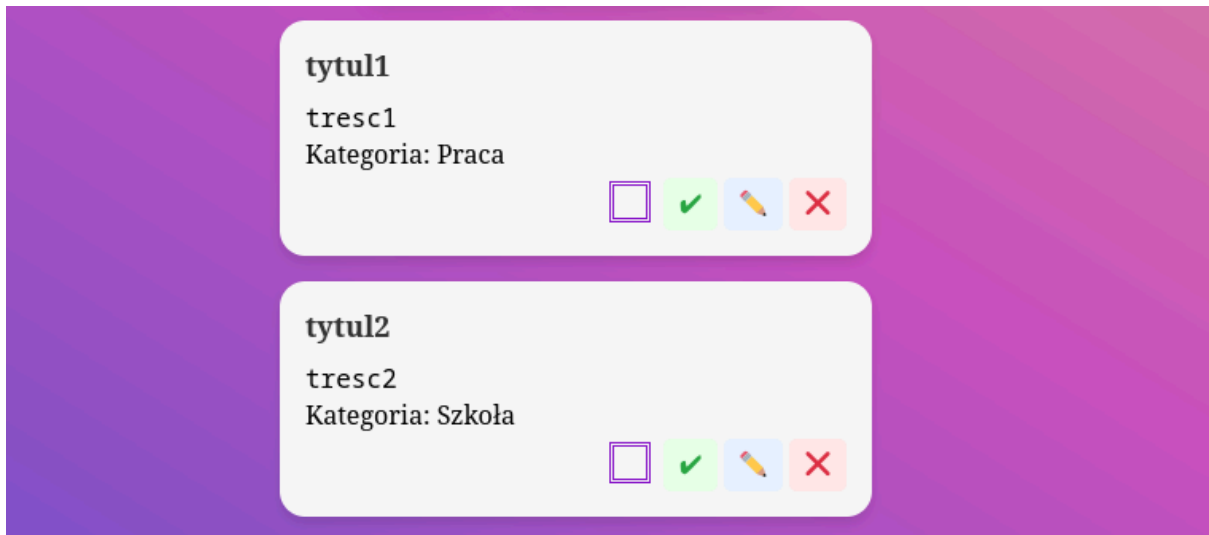
- "Dodaj" - dodaje nową notatkę lub aktualizuje edytowaną.
- "Usuń wszystko" - usuwa wszystkie zapisane notatki.

Dodaj

Usuń wszystko

6. Lista zapisanych notatek

- Miejsce, w którym wyświetlane są wszystkie utworzone notatki.



Plik: **script.js**

Plik JavaScript odpowiada za interaktywność aplikacji oraz logikę zarządzania notatkami.

Najważniejsze funkcje:

1. Funkcje formatowania tekstu

- **formatText(command)** - Umożliwia stosowanie stylów (pogrubienie, kursywa itp.) na zaznaczonym tekście.

```
* Funkcja formatuje tekst w edytorze notatki.
*
* Umożliwia zastosowanie formatowania tekstu (np. pogrubienie, kursywa, itp.) w edytorze
* Funkcja wykonuje odpowiednią komendę formatowania na zaznaczonym fragmencie tekstu
* w edytorze, który jest polem o identyfikatorze 'noteInput'.
*
* @param {string} command - Komenda formatowania do wykonania np. pogrubienia, kursywy
*/
function formatText(command) {
  // Ustawienie fokusu na pole edycji notatki (noteInput)
  document.getElementById('noteInput').focus();
  // Wykonanie komendy formatowania na zaznaczonym fragmencie tekstu
  document.execCommand(command, false, null);
}
```

- **changeTextColor(color)** - Zmienia kolor zaznaczonego tekstu.

```
* Zmienia kolor wybranego tekstu na wskazany kolor.
* @param {string} color - Kolor, który ma zostać zastosowany do wybranego tekstu.
*/
function changeTextColor(color) {
  document.getElementById('noteInput').focus();
  document.execCommand('foreColor', false, color);
}
```

- **applyBulletPoint()** - Dodaje wypunktowanie do zaznaczonego tekstu.

```

* Opakowuje wybrany tekst w punkt listy (bullet point).
* Tworzy nową listę nieuporządkowaną (<ul>) i wstawia wybrany tekst do elementu <li>.
*/
function applyBulletPoint() {
  document.getElementById('noteInput').focus();
  const selectedText = window.getSelection();
  const range = selectedText.getRangeAt(0);
  const bulletPointList = document.createElement('ul');
  const listItem = document.createElement('li');
  listItem.innerHTML = range.toString();
  bulletPointList.appendChild(listItem);
  range.deleteContents();
  range.insertNode(bulletPointList);
}

```

2. Zarządzanie notatkami

- o `addOrUpdateNote()` - Dodaje nową notatkę lub aktualizuje istniejącą.

```

function addOrUpdateNote() {

  // Pobranie wartości tytułu notatki z pola tekstowego

  const noteTitle = document.getElementById('noteTitle').value.trim(); //
  // Nadawanie tytułu

  // Pobranie zawartości notatki z edytowalnego pola

  const noteInput = document.getElementById('noteInput').innerHTML;

  // Pobranie wybranej kategorii z listy rozwijanej

  const categorySelect = document.getElementById('categorySelect');

  const customCategoryInput =
  document.getElementById('customCategoryInput');

  // Ustalenie kategorii notatki: jeśli użytkownik wybrał 'custom', używamy
  // wartości z pola tekstowego

  let category = categorySelect.value === 'custom' ?
  customCategoryInput.value.trim() : categorySelect.value;

  // Sprawdzenie, czy zawartość notatki jest pusta

  if (!noteInput) {

    alert("Nie wpisano tekstu notatki.");

    return;

  }

  // Sprawdzenie, czy wybrano kategorię

  if (!category) {

```

```
        alert("Proszę wybrać kategorię przed zapisaniem notatki.");

        return;
    }

    // Sprawdzenie, czy tytuł notatki jest pusty
    if (!noteTitle) {

        alert("Proszę wpisać tytuł notatki.");

        return;
    }

    // Pobranie listy notatek, do której dodamy nową notatkę lub
    zaktualizujemy istniejącą

    const noteList = document.getElementById('noteList');

    // Sprawdzenie, czy edytujemy istniejącą notatkę
    if (editingNoteCard) {

        // Zaktualizowanie tytułu, treści i kategorii istniejącej notatki

        const noteCardTitle =
            editingNoteCard.querySelector('.note-card-title');

        const noteCardContent =
            editingNoteCard.querySelector('.note-card-content');

        const noteCardCategory =
            editingNoteCard.querySelector('.note-card-category');

        // Ustawienie nowych wartości dla tytułu, treści i kategorii notatki
        noteCardTitle.textContent = noteTitle; // Ładowanie tytułu
        noteCardContent.innerHTML = noteInput;
        noteCardCategory.textContent = `Kategoria: ${category}`;

        // Po zakończeniu edycji, resetowanie zmiennej `editingNoteCard`
        editingNoteCard = null;
    } else {

        // Tworzenie nowej notatki, jeśli nie edytujemy istniejącej

        const noteCard = document.createElement('div');
```

```
noteCard.classList.add('note-card');

// Tworzenie i ustawianie tytułu nowej notatki
const noteCardTitle = document.createElement('div');
noteCardTitle.classList.add('note-card-title');
noteCardTitle.textContent = noteTitle; // Ładowanie tytułu

// Tworzenie i ustawianie treści nowej notatki
const noteCardContent = document.createElement('div');
noteCardContent.classList.add('note-card-content');
noteCardContent.innerHTML = noteInput;

// Tworzenie i ustawianie kategorii nowej notatki
const noteCardCategory = document.createElement('div');
noteCardCategory.classList.add('note-card-category');
noteCardCategory.textContent = `Kategoria: ${category}`;

// Tworzenie kontenera akcji dla notatki
const actionsContainer = createActionContainer(noteCard);

// Dodanie wszystkich elementów (tytuł, treść, kategoria, akcje) do
nowej notatki
noteCard.appendChild(noteCardTitle); // Ładowanie tytułu
noteCard.appendChild(noteCardContent);
noteCard.appendChild(noteCardCategory);
noteCard.appendChild(actionsContainer);

// Dodanie nowej notatki do listy notatek
noteList.appendChild(noteCard);
}

// Resetowanie formularza: czyszczenie pól tytułu, treści, kategorii i
ustawienie domyślnej wartości kategorii
document.getElementById('noteTitle').value = '';
```

```

document.getElementById('noteInput').innerHTML = '';

customCategoryInput.value = '';

categorySelect.value = 'Praca';

customCategoryInput.style.display = 'none';

// Zapisanie notatek (np. w localStorage lub w innym miejscu)

saveNotes();

// Dostosowanie wysokości pola edycji notatki, aby pasowała do zawartości
adjustTextareaHeight(document.getElementById('noteInput'));
}

```

- o **editNote(noteCard)** - Przenosi dane z wybranej notatki do formularza edycji, umożliwiając ich modyfikację.

```

function editNote(noteCard) {

    // Ustawienie zmiennej `editingNoteCard`, która przechowuje
    // odniesienie do edytowanej notatki

    editingNoteCard = noteCard;

    // Pobranie tytułu notatki z elementu HTML i przypisanie go do pola
    // edycji

    const noteTitle =
noteCard.querySelector('.note-card-title').textContent;

    // Pobranie zawartości notatki (z zachowaniem linii przerwy) i
    // przypisanie jej do pola edycji

    const noteContent =
noteCard.querySelector('.note-card-content').innerHTML.replace(/<br>/g,
'\n'); // Zamiana <br> na \n

    // Pobranie kategorii notatki i usunięcie prefiksu "Kategoria: "

    const noteCategory =
noteCard.querySelector('.note-card-category').textContent.replace('Kate
goria: ', '');

    // Ustawienie tytułu i zawartości notatki w formularzu edycji

    document.getElementById('noteTitle').value = noteTitle; //
    Ustawienie tytułu w polu tekstowym

```

```

    document.getElementById('noteInput').innerHTML =
noteContent.replace(/\n/g, '<br>'); // Ustawienie treści w polu edycji
(z powrotem zamiana \n na <br>)

    // Pobranie elementów formularza kategorii

    const categorySelect = document.getElementById('categorySelect');

    const customCategoryInput =
document.getElementById('customCategoryInput');

    // Sprawdzenie, czy kategoria jest standardowa, czy niestandardowa

    if (['Praca', 'Szkoła', 'Zakupy', 'Do
zrobienia'].includes(noteCategory)) {

        // Jeśli kategoria jest standardowa, ustawić odpowiednią
kategorię w formularzu i ukryć pole dla niestandardowej kategorii

        categorySelect.value = noteCategory;

        customCategoryInput.style.display = 'none';

    } else {

        // Jeśli kategoria jest niestandardowa, ustawić kategorię na
'custom' i pokazać pole do wprowadzenia własnej kategorii

        categorySelect.value = 'custom';

        customCategoryInput.value = noteCategory;

        customCategoryInput.style.display = 'block';

    }

    // Dostosowanie wysokości pola edycji notatki, aby pasowała do
zawartości

    adjustTextareaHeight(document.getElementById('noteInput'));
}

```

- `clearNotes()` - Usuwa wszystkie zapisane notatki po potwierdzeniu przez użytkownika.


```

* Funkcja umożliwia usunięcie wszystkich notatek.
*
* Po wywołaniu tej funkcji użytkownik zostanie zapytany o potwierdzenie usunięcia wszystkich notatek
* Jeśli użytkownik potwierdzi, wszystkie notatki zostaną usunięte.
*/
function clearNotes() {
  // Wyświetlenie okna dialogowego z zapytaniem o potwierdzenie usunięcia wszystkich notatek
  if (confirm("Czy na pewno chcesz usunąć wszystkie notatki?")) {
    // Usunięcie wszystkich elementów w kontenerze notatek
    document.getElementById('noteList').innerHTML = '';

    // Usunięcie danych notatek zapisanych w localStorage
    localStorage.removeItem('notes');
  }
}

```

- `saveNotes()` - Zapisuje notatki w lokalnej pamięci przeglądarki.

```

* Funkcja zapisuje wszystkie notatki w lokalnej pamięci przeglądarki (localStorage).
* Przechodzi przez wszystkie notatki wyświetlane na stronie, zbiera ich dane (tytuł, treść, kategoria, status przypięcia, status wykonania),
* a następnie zapisuje je w 'localStorage' w formacie JSON, aby zachować notatki po odświeżeniu strony.
*/
function saveNotes() {
  const noteList = document.getElementById('noteList');
  const notes = [];

  for (const noteCard of noteList.children) {
    const noteTitle = noteCard.querySelector('.note-card-title').textContent;
    const noteContent = noteCard.querySelector('.note-card-content').innerHTML;
    const noteCategory = noteCard.querySelector('.note-card-category').textContent.replace('Kategoria: ', '');
    const isChecked = noteCard.querySelector('.note-card-pinned-status input').checked;
    const isDone = noteCard.querySelector('.note-card-content').classList.contains('note-card-done');
    notes.push({ title: noteTitle, content: noteContent.replace(/<br>/g, '\n'), category: noteCategory, checked: isChecked, done: isDone });
  }

  localStorage.setItem('notes', JSON.stringify(notes));
}

```

3. Interfejs użytkownika

- `adjustTextareaHeight(textarea)` - Dostosowuje wysokość pola edycji do zawartości.

```

* Aktualizuje stan paska narzędzi na podstawie bieżącego zaznaczenia tekstu.
*/
document.addEventListener('selectionchange', updateToolbarState);
editingNoteCard = null;

adjustTextareaHeight(document.getElementById('noteInput'));
function adjustTextareaHeight(textarea) {
  textarea.style.height = 'auto';
  const computedStyle = window.getComputedStyle(textarea);
  const lineHeight = parseFloat(computedStyle.lineHeight);
  textarea.style.height = (textarea.scrollHeight - lineHeight * 2) + 'px';
}

document.getElementById('categorySelect').addEventListener('change', function () {
  const customCategoryInput = document.getElementById('customCategoryInput');
  customCategoryInput.style.display = this.value === 'custom' ? 'block' : 'none';
});

```

- `updateToolbarState()` - Aktualizuje stan przycisków paska narzędzi w zależności od zastosowanego formatowania tekstu.

```

* Funkcja aktualizuje stan paska narzędzi w oparciu o aktualnie zaznaczony tekst w edytorze.
*
* Funkcja sprawdza, czy zaznaczenie tekstu znajduje się w polu edycji notatki ('noteInput').
* Jeśli tak, to na podstawie aktualnego formatowania tekstu (np. pogrubienie, kursywa, podkreślenie, przekreślenie),
* aktualizowane są przyciski narzędzi na pasku narzędziowym, przypisując im klasę 'active', jeśli dany styl jest aktywny.
* Dzięki temu użytkownik widzi, które style są aktualnie zastosowane na zaznaczonym fragmencie tekstu.
*
* @returns {void}
*/

function updateToolbarState() {

    // Pobranie elementu edytora notatki
    const noteInput = document.getElementById('noteInput');

    // Pobranie zaznaczenia użytkownika
    const selection = window.getSelection();

    // Sprawdzanie, czy zaznaczenie znajduje się w obrębie elementu 'noteInput'
    const isWithinNoteInput = noteInput.contains(selection.anchorNode);

    // Jeśli zaznaczenie nie jest w 'noteInput', nie aktualizujemy stanu paska narzędzi
    if (!isWithinNoteInput) return;

    // Sprawdzenie stanu formatowania zaznaczonego tekstu
    const isBold = document.queryCommandState('bold');
    const isItalic = document.queryCommandState('italic');
    const isUnderline = document.queryCommandState('underline');
    const isStrikeThrough = document.queryCommandState('strikeThrough');

    // Aktualizacja paska narzędzi - dodanie lub usunięcie klasy 'active' w zależności od stanu formatowania
    document.querySelector('[onclick="formatText(\'bold\')"]').classList.toggle('active', isBold);
    document.querySelector('[onclick="formatText(\'italic\')"]').classList.toggle('active', isItalic);
    document.querySelector('[onclick="formatText(\'underline\')"]').classList.toggle('active', isUnderline);
    document.querySelector('[onclick="formatText(\'strikeThrough\')"]').classList.toggle('active', isStrikeThrough);
}

```

4. Sortowanie notatek

- `sortNotes(criteria)` - Sortuje notatki według wybranego kryterium (np. tytuł, kategoria, przypięte notatki).

```
* Funkcja umożliwia sortowanie notatek zapisanych w notatniku.
* Sortowanie odbywa się według kryterium podanego przez użytkownika.
* Możliwe kryteria:
* - "title": sortuje alfabetycznie (A-Z) według tytułów notatek.
* - "category": sortuje alfabetycznie (A-Z) według nazw kategorii.
* - "checked": sortuje notatki z przypiętym statusem jako pierwsze,
*   a następnie alfabetycznie według tytułów w każdej grupie.
* @param {string} criteria - Kryterium sortowania ("title", "category", "checked").
*/
function sortNotes(criteria) {
    // Pobiera listę notatek z elementu DOM
    const noteList = document.getElementById('noteList');
    const notes = Array.from(noteList.children);

    // Sortuje notatki na podstawie wybranego kryterium
    notes.sort((a, b) => {
        // Sortowanie alfabetyczne A-Z według tytułów notatek
        if (criteria === 'title') {
            return a.querySelector('.note-card-title').textContent.localeCompare(
                b.querySelector('.note-card-title').textContent
            );
        }
        // Sortowanie alfabetyczne A-Z według nazw kategorii
        } else if (criteria === 'category') {
            return a.querySelector('.note-card-category').textContent.localeCompare(
                b.querySelector('.note-card-category').textContent
            );
        }
        // Sortowanie, gdzie przypięte notatki są wyświetlane jako pierwsze
        } else if (criteria === 'checked') {
            // Sprawdza status "przypięte" dla obu notatek
            const aChecked = a.querySelector('.note-card-pinned-status input').checked;
            const bChecked = b.querySelector('.note-card-pinned-status input').checked;

            // Sortuje przypięte notatki jako pierwsze
            if (aChecked !== bChecked) {
                return bChecked - aChecked;
            }

            // Jeśli oba statusy są takie same, sortuje alfabetycznie według tytułu
            return a.querySelector('.note-card-title').textContent.localeCompare(
                b.querySelector('.note-card-title').textContent
            );
        }
    });
};
```

5. Dodatkowe funkcje

- `createActionsContainer(noteCard)` - Tworzy kontener akcji dla notatki (przypinanie, edytowanie, usuwanie itp.).

```
* Funkcja tworzy kontener akcji dla notatki, umożliwiający przypinanie, edytowanie, oznaczanie jako wykonane oraz usuwanie notatki.
*/
function createActionsContainer(noteCard) {
  const actionsContainer = document.createElement('div');
  actionsContainer.classList.add('note-card-actions');

  // BK
  // umożliwia użytkownikowi przypięcie notatki za pomocą checkboxa
  // checkbox dodawany jest do każdej tworzonej notatki
  const pinNoteContainer = document.createElement('label');
  pinNoteContainer.classList.add('note-card-pinned-status');

  const pinNoteCheckbox = document.createElement('input');
  pinNoteCheckbox.type = 'checkbox';
  pinNoteCheckbox.classList.add('note-card-pinned');
  pinNoteCheckbox.onchange = saveNotes;
  pinNoteCheckbox.checked = false;

  pinNoteContainer.textContent = ' ';
  pinNoteContainer.prepend(pinNoteCheckbox);
  // BK

  const checkButton = document.createElement('span');
  checkButton.classList.add('note-card-check');
  checkButton.textContent = '✓';
  checkButton.onclick = () => toggleDone(noteCard);

  const editButton = document.createElement('span');
  editButton.classList.add('note-card-edit');
  editButton.textContent = '✎';
  editButton.onclick = () => editNote(noteCard);

  const deleteButton = document.createElement('span');
  deleteButton.classList.add('note-card-delete');
  deleteButton.textContent = '✖';
  deleteButton.onclick = () => deleteNote(noteCard);

  actionsContainer.appendChild(pinNoteContainer);
  actionsContainer.appendChild(checkButton);
  actionsContainer.appendChild(editButton);
  actionsContainer.appendChild(deleteButton);

  return actionsContainer;
}
```

- `toggleDone(noteCard)` - Oznacza notatkę jako wykonaną.

```
* Funkcja przełącza status wykonania notatki.
* Gdy notatka jest oznaczona jako "do zrobienia", funkcja zmienia jej stan na "zrobione" i odwrotnie.
* Funkcja dodaje lub usuwa klasę 'note-card-done' z treści notatki, co może zmieniać jej wygląd (np. przez przekreślenie tekstu)
* Po zmianie statusu wykonania, zapisuje zmodyfikowane notatki w localStorage.
*/
function toggleDone(noteCard) {
  const noteContent = noteCard.querySelector('.note-card-content');
  noteContent.classList.toggle('note-card-done');
  saveNotes();
}
```

Struktura CSS (plik **styles.css**)

Plik **styles.css** odpowiada za wygląd aplikacji oraz zapewnia responsywność. Zawiera następujące sekcje:

1. Style ogólne

- Ustawienia podstawowe:

```
* Ogólne style dla dokumentu.
* Tło strony ma gradient przechodzący od niebieskiego, przez różowy, do żółtego.
* Elementy na stronie są rozmieszczane w kolumnie i centrowane.
*/
html, body {
  min-height: 100%;
}

body {
  background-color: #4158D0;
  background-image: linear-gradient(32deg, #4158D0 0%, #C850C0 46%, #FFCC70 100%);
  display: flex;
  flex-direction: column;
  align-items: center;
}
```

- Stylizacja nagłówka:

```
* Ogólne stylowanie nagłówka h1.
*/
h1 {
  font-size: 4rem;
  font-weight: bold;
  font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
  -webkit-text-stroke-width: 0.1px;
  -webkit-text-stroke-color: beige;
  color: black;
  text-shadow: 1px 1px 3px rgba(0, 0, 0, 0.2);
  margin-bottom: 20px;
}
```

2. Style interaktywne

- Przyciski paska narzędzi:

```
/* Stylizacja przycisków na pasku narzędziowym z animacjami przejścia.
 */
.toolbar button {
  width: 36px;
  height: 36px;
  display: flex;
  align-items: center;
  justify-content: center;
  border: none;
  background-color: #fff;
  border-radius: 4px;
  cursor: pointer;
  font-size: 16px;
  transition: background-color 0.2s, transform 0.1s;
  box-shadow: 0 1px 3px rgba(0, 0, 0, 0.1);
}

/*
 * Zmiana tła i efekt powiększenia przycisku po najechaniu.
 */
.toolbar button:hover {
  background-color: #f0f0f0;
  transform: scale(1.1);
}

/*
 * Stylizacja aktywnego przycisku na pasku narzędziowym. Zmienia tło i dodaje obramowanie.
 */
.toolbar button.active {
  background-color: #ddd;
  border: 1px solid #aaa;
}
```

- Podświetlanie aktywnego przycisku:

```
/* Efekt zmiany tła przycisku po najechaniu
 */
button:hover {
  background-color: #ddd;
}
```

3. Lista notatek

- Notatki wyświetlane w formie kart:

```
* Stylizacja karty notatki.
*/
.note-card {
  display: block;
  width: 340px;
  background-color: #f8f8f8;
  border-radius: 16px;
  box-shadow: 0px 4px 6px rgba(0, 0, 0, 0.1);
  padding: 16px;
  margin-top: 16px;
  margin-bottom: 16px;
  word-wrap: break-word;
  white-space: pre-wrap;
}
```

- Akcje notatek (przypinanie, edycja, usuwanie):


```
* Stylizacja przycisków akcji karty notatki: usuwanie, edytowanie i zaznaczanie.
* Każdy przycisk ma inny kolor i tło, zmieniające się przy najechaniu.
*/
.note-card-delete,
.note-card-edit,
.note-card-check {
  cursor: pointer;
  color: #ff4d4d;
  font-weight: bold;
  padding: 5px 10px;
  border-radius: 5px;
  background-color: #ffe6e6;
  transition: background-color 0.3s;
}
```

Instrukcja użytkowania


1. Tworzenie nowej notatki

- Wprowadź tytuł, treść i wybierz kategorię. Możesz użyć paska narzędzi do formatowania tekstu.
- Kliknij przycisk "Dodaj", aby zapisać notatkę.

2. Edycja istniejącej notatki

- Kliknij ikonę  na notatce, aby przenieść jej dane do formularza.
- Wprowadź zmiany i kliknij "Dodaj", aby zaktualizować notatkę.

3. Usuwanie notatek

- Kliknij  na konkretnej notatce, aby ją usunąć.
- Użyj przycisku "Usuń wszystko", aby usunąć wszystkie notatki.

4. Sortowanie notatek

- Użyj paska sortowania, aby uporządkować notatki według wybranego kryterium.

Dalsze ulepszenia

- Dodanie obsługi daty tworzenia i edycji notatek.
- Możliwość eksportu notatek do pliku (np. PDF, TXT).
- Integracja z backendem w celu synchronizacji danych między urządzeniami.

Podsumowanie

Aplikacja "Notatnik" to narzędzie wspierające organizację i zarządzanie codziennymi notatkami. Dzięki prostocie obsługi oraz funkcjom formatowania tekstu stanowi praktyczne rozwiązanie dla użytkowników potrzebujących szybkiego dostępu do zapisanych informacji.