TaskBoard

(Task & Notes Dashboard)

Technická dokumentace

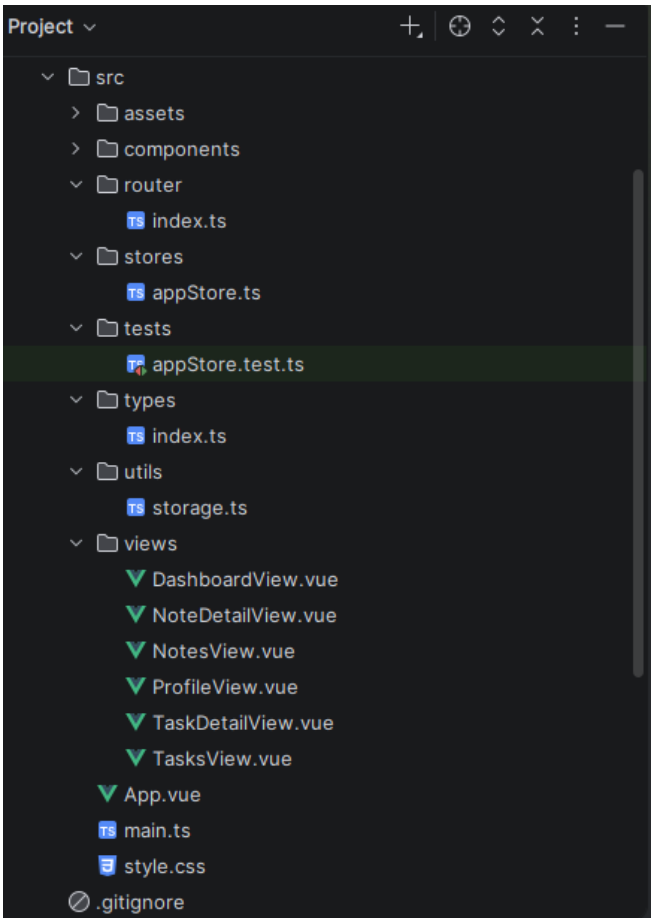
Adam Zinzer

1. Použitý framework a technologie

- **Vue 3** (Composition API)
- **Vite** – build a vývojový nástroj
- **Pinia** – správa globálního stavu aplikace
- **Vue Router** – SPA routování
- **TypeScript** – typová bezpečnost
- **LocalStorage** – simulace backendového úložiště
- **Vitest** – jednotkové testování
- **Cypress** – end-to-end testování
- **Chrome DevTools & Lighthouse** – profilování výkonu

2. Architektura aplikace

Aplikace je navržena jako SPA s jasným oddělením jednotlivých vrstev:



Struktura projektu (zjednodušeně)

Views

Každé hlavní zobrazení aplikace je implementováno jako samostatná view komponenta. Navigace mezi nimi je řešena pomocí Vue Routeru bez obnovy stránky.

Aplikace obsahuje minimálně **6 logicky oddělených zobrazení**:

- Dashboard
- Tasks
- Task Detail
- Notes
- Note Detail
- Profile

3. Správa stavu a práce s daty

Pinia Store

Globální stav aplikace je spravován pomocí **Pinia**, kde jsou uchovávány:

- seznam úkolů (tasks)
- seznam poznámek (notes)

Každý úkol obsahuje:

- id
- title
- description (volitelné)
- completed
- order

Ukládání dat

Data jsou ukládána do **LocalStorage**, čímž je simulována backendová vrstva.
Při každé změně dat (přidání, odebrání, změna stavu) je stav automaticky synchronizován s LocalStorage.

Validace vstupů

- povinné pole title při vytváření úkolů a poznámek
- základní ošetření prázdných vstupů na straně klienta

4. Použité nástroje a Testování

Jednotkové testy

Pro jednotkové testování byl použit framework **Vitest**, který je kompatibilní s Vite a nabízí API podobné Jestu.

Testována byla aplikační logika uložená ve store (Pinia):

- přidání úkolu
- změna stavu úkolu
- odebrání úkolu
- přidání poznámky
- odebrání poznámky

UI komponenty testovány nebyly, jelikož hlavní logika aplikace je centralizována ve store.

End-to-End testy

Pro simulaci reálného chování uživatele byl použit **Cypress**.
Byl vytvořen jednoduchý e2e test, který:

- otevře aplikaci
- přejde na stránku Tasks
- přidá nový úkol
- ověří jeho zobrazení v seznamu

Použité nástroje pro potimalizaci výkonu

1) Lighthouse

Pro analýzu výkonu aplikace byl použit nástroj **Lighthouse** integrovaný v Chrome DevTools. Analýza byla provedena v desktop režimu a aplikace dosahuje dobrého skóre výkonu.

2) Chrome DevTools

Pomocí panelu **Performance** bylo sledováno chování aplikace při:

- navigaci mezi jednotlivými views
- práci s daty (přidání a změna úkolu)

Nebyl zaznamenán žádný výrazný výkonnostní problém.