GYMNASIUM JANA KEPLERA

Parléřova 2/118, 169 00 Praha 6

Školní webová stránka

Maturitní práce

Autor: David Frederick Batt

Třída: 4.A

Školní rok: 2024/2025

Předmět: Informatika

Vedoucí práce: Ing. Bc. Jan Matoušek

Praha, 2025

Gymnázium Jana Keplera

Kabinet informatiky

ZADÁNÍ MATURITNÍ PRÁCE

- *Student:* David Frederick Batt
- *Třída:* 4.A
- *Školní rok:* 2024/2025
- *Vedoucí práce:* Jan Matoušek
- *Název práce:* Frontend nového školního webu GJK
- URL repozitáře: https://github.com/da-batt/GJK-web

Pokyny pro vypracování:

Cíle
Vytvořit frontend pro web, který by mohl sloužit jako náhrada aktuálního školního webu GJK. Součástí bude i admin zóna pro spravování uživatelů a zveřejňování aktualit.

Implementace
Frontend webu bude zhotoven pomocí Next.js. Data budou skladována a načítána z vlastního rust backendu, který není součástí tohoto maturitního projektu, avšak jehož zdrojový kód je součástí repozitáře.

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů. Nemám žádné námitky proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Praze dne 2. března 2025 David Frederick Batt

Poděkování

Chtěl bych vřele poděkovat Ing. Bc. Janu Matouškovi za vedení mého projektu, ochotu mi vždy poskytnout zpětnou vazbu a poskytnutí možnosti se na tomto projektu podílet. Děkuji také panu řediteli, Mgr. Karlu Žďárkovi, PhD., který neváhal se mnou projekt konzultovat a vždy si ma mne udělal čas. Zároveň bych chtěl poděkovat školnímu ICT administrátorovi, Ing. Martinovi Wohankovi, za kvalitní zpětnou vazbu a konzultace.

Abstrakt

Tato maturitní práce se zaměřuje na návrh a implementaci frontendu nového školního webu Gymnázia Jana Keplera. Cílem projektu bylo vytvořit moderní, přehledný a snadno spravovatelný web, který by mohl nahradit stávající školní stránky. Frontend byl vyvinut pomocí frameworku Next.js, přičemž pro správu obsahu byl využit headless CMS Payload. Výsledkem je webová aplikace optimalizovaná pro výkon, uživatelskou přívětivost a jednoduchou administraci obsahu.

Klíčová slova

frontend, Next.js, školní web, headless CMS, Payload CMS, webová aplikace

Abstract

This thesis focuses on the design and implementation of the frontend for the new school website of Gymnázium Jana Keplera. The goal of the project was to create a modern, user-friendly, and easily manageable website that could replace the current school site. The frontend was developed using the Next.js framework, while the content management was handled through the Payload headless CMS. The result is a web application optimized for performance, usability, and seamless content administration.

Keywords

frontend, Next.js, school website, headless CMS, Payload CMS, web application

Obsah

1. Teoretická část

1.1 Popis problému

Školní webové stránky jsou nezbytným nástrojem pro sdílení důležitých informací, jako jsou aktuality, rozvrhy hodin, kontakty či dokumenty. Slouží nejen studentům a učitelům, ale i rodičům a širší veřejnosti, a proto je důležité, aby byly přehledné, rychlé a snadno použitelné. Pokud web tyto požadavky nesplňuje, může negativně ovlivňovat uživatelskou zkušenost.

Nutnost vytvořit novou školní webovou stránku vychází z nespokojenosti s aktuálním řešením, kterou vyjádřil ředitel Gymnázia Jana Keplera (GJK), Mgr. Karel Žďárek, PhD., a další zainteresované osoby. Současný web působí vizuálně zastarale a nepřehledně, což ztěžuje orientaci uživatelů a negativně ovlivňuje vnímání školy. Dalším problémem je náročná správa obsahu – současný systém založený na WordPress je složitý na údržbu, což způsobuje komplikace při aktualizaci informací. Tyto faktory vedly k rozhodnutí vytvořit nový, moderní a uživatelsky přívětivý web.

1.2 Požadavky a cíle

Na základě opakovaných konzultací s panem ředitelem Mgr. Karlem Žďárkem, PhD. a ICT administrátorem Ing. Martinem Wohankou byly pro novou webovout stránku stanoveny konkrétní požadavky.

Mezi hlavní požadavky patří:

- Přehlednost a uživatelská přívětivost – Webová stránka by měla být intuitivní a snadno navigovatelná pro všechny uživatele, včetně studentů, rodičů, pedagogů i dalších návštěvníků.
- Grafické webové rozhraní pro správu obsahu – Aktuality a stránky musí být upravitelné oprávněnými uživateli bez nutnosti hlubších technických znalostí.
- Možnost provozu na školním serveru – Webová stránka musí být plně hostovatelná na vlastním serveru gymnázia, bez závislosti na externích poskytovatelech hostingu.
- Moderní vzhled – Design nové stránky by měl odpovídat současným standardům a minimalizovat zbytečný vizuální ruch.

2. Implementace

2.1 Architektura

Původně měla na projekt být aplikována tradiční třívrstvá architektura, tj. prezentační vrstva (frontend), aplikační vrstva (API/backend) a datová vrstva (databáze). Vlastnoručně vytvořený backend psaný v praxi však znamenal, že velké množství úsilí bylo stráveno na implementaci administračního rozhraní a webová aplikace byla nakonec komplexní a nedostatečně flexibilní pro budoucí správu. Nakonec byl tedy backend nahrazen za headless systém pro správu obsahu (CMS), který splňuje stejné funkce jako předešlé řešení. Tento přístup umožnil zcela oddělit správu obsahu od zbytku frontendu, který mohl být zachován, což ve výsledku bylo mnohem čistší a lépe spravovatelné řešení pro tento projekt. Zároveň je pro projekt nezbytně důležité, aby správa obsahu fungovala bez problémů v produkci a tak dávalo smysl využít známé a otestované řešení místo složitého na míru vytvořeného systému, obzvlášť když vezmeme v potaz, že to původně neměla být hlavní část tohoto maturitního projektu.

2.2 Design

Přístup k designu stránky byl v průběhu implementace změněn. Ze začátku měl být podle určení pana ředitele použitý návrh vytvořen třetí stranou, brzy se však prokázalo, že neodpovídal stanoveným požadavkům. Design stránky byl tedy nakonec zastřešen do tohoto projektu. Průběh designového procesu pro tento projekt probíhal iterativně, kdy po každé iteraci byl návrh scvhálen pane ředitelem. Proces byl rozdělen do následujících kroků:

1. Stanovení požadavků
2. Tvorba sitemap a wireframů
3. Vizuální design
4. Testování + návrat ke kroku 3. pokud zapotřebí

Vzhledem k tomu, že design webu nebyl předpokládanou součástí tohoto maturitního projektu, tak byl celý designový proces trochu problematický. Během tvorby tohoto projektu chyběla jasná vize toho, jak má přesně nová stránka vypadat a co má obsahovat. Ve výsledku se pracovalo hlavně s výplňkovým obsahem a neutrální vizuální identitou. Design tohoto projektu mohl rozhodně být rozvržen a zpracován lépe.

2.3 Výběr technologií

2.3.1 Frontend

Při výběru frontendu byly alternativy vybírány na základě vlastností, které jsou následující:

- Technologie umožňují snadné a efektivní propojení s headless CMS
- Nabízejí klíčové funkce jako server-side rendering (SSR) a statickou generaci stránek (SSG) pro lepší výkon a SEO.
- Mají rozsáhlou a aktivní vývojářskou komunitu, což zajišťuje dlouhodobou podporu, kvalitní dokumentaci a dostupnost knihoven a nástrojů.
- Jsou moderní a odpovídají ostatním požadavkům projektu.

Framework	**Typ frameworku**	**Podpora SSR**	**Podpora SSG**
Next.js	Založený na Reactu	Ano	Ano
Gatsby	Založený na Reactu	Omezená	Ano
Nuxt.js	Založený na Vue	Ano	Ano
SvelteKit	Založený na Svelte	Ano	Ano
Angular Universal	Založený na Angular	Ano	Omezená

Tabulka 2.1: Přehled zvážených alternativ pro frontend

Pro tvorbu frontendu byl z alternativ zvolen Next.js, protože, jak je patrné z tabulky 2.1, podporuje SSR i SSG a zároveň je založený na knihovně React, která má ve srovnání s Vue, Svelte a Angular největší komunitu. Tato rozsáhlá komunita zaručuje dlouhodobou podporu, rychlé řešení případných problémů a širokou dostupnost knihoven, nástrojů a zdrojů. Silná komunita také znamená, že framework bude pravděpodobně dobře podporován i v budoucnu, což je zásadní pro udržitelnost a rozšiřitelnost projektu. Další důležitá funkce Next.js pro tento projekt je automatický routing, který usnadňuje organizaci kódu. Stačí umístit soubor do složky `app/` a Next.js z něj automaticky vytvoří odpovídající URL adresu, čímž se výrazně zjednodušuje tvorba webu.

2.3.2 Styly

Pro styly byl z nalezených alternativ uvedených v tabulce 2.2 zvolen Tailwind CSS, protože nabízí vysokou flexibilitu a efektivitu díky utility-first přístupu, což znamená, že místo práce s předdefinovanými komponentami, jako v Bootstrap a Bulma, umožňuje vytvářet vlastní styly pomocí malých, konkrétních CSS tříd. Tento přístup poskytuje vysokou kontrolu nad designem školního webu,

Technologie	Výhody	Nevýhody
Čisté CSS / SCSS	Plná kontrola nad vzhledem, optimální výkon pro menší projekty	Více psaní kódu, obtížnější údržba
Bootstrap	Rychlý vývoj díky hotovým komponentám	Méně přizpůsobitelný, standardní vzhled, velké množství nevyužitého kódu
Tailwind CSS	Rychlý a flexibilní utility-first přístup, optimalizováno pro výkon	Množství tříd může vést k hůře čitelnému markupu
Bulma	Hotové komponenty, flexibilní layout	Méně pokročilých komponent než Bootstrap, omezená podpora pro interaktivní komponenty

Tabulka 2.2: Porovnání zvážených alternativ pro tvorbu stylů

ale zároveň usnadňuje rychlou tvorbu responzivních a přizpůsobitelných rozložení oproti čistému CSS/SCSS. Tailwind také v produkci automaticky odstraní všechny nepoužité soubory CSS, což znamená, že konečný CSS bundle je nejmenší možný.

2.3.3 Headless CMS

Při výběru headless CMS, musela řešení splňovat podmínku, že jsou hostovatelné na vlastním serveru na základě požadavků stanovených výše. Podle analýzy alternativ uvedených v tabulce 2.3 byl pro projekt zvolen Payload CMS, protože zprostředkovává veškerou potřebnou funkcionalitu pro projekt bez potřeby pro rozšíření, je v porovnání s alternativami nejvýkonnější a zároveň je postaven na Next.js, což přináší pro projekt řadu výhod. Jednou takovou výhodou je ta, že může CMS běžet spolu s frontendem v jednom procesu, čímž je deployment zjednodušen. Zároveň CMS zprostředkovává lokální API, díky kterému lze integrovat s databází přímo na serveru, snižujíc latenci. CMS je také nakonfigurován v kódu, což je z vývojářského hlediska mnohem přívětivější než alternativy, které se konfigurují přes grafické webové rozhraní.

CMS	Relativní výkon [1]	Konfigurace	Ekosystém	API a integrace
Strapi	Nejnižší	GUI	Velké množství pluginů a integrací	REST, GraphQL
Payload CMS	Nejvyšší	Konfigurační soubor	Omezená nabídka rozšíření	REST, GraphQL, integrace s Next.js
Directus	Střední	GUI	Střední množství rozšíření	REST, GraphQL

Tabulka 2.3: Přehled zvážených alternativ pro headless CMS

2.3.4 TypeScript

V tomto projektu je použití TypeScript jako hlavního jazyka vhodnou volbou, protože jej podporují všechny ostatní používané technologie a zároveň přináší významné výhody oproti JavaScriptu. Nejzásadnějším rozdílem je statické typování, které pomáhá odhalit chyby již během psaní kódu, čímž zvyšuje jeho spolehlivost a usnadňuje refaktorování. Na rozdíl od JavaScriptu umožňuje TypeScript definovat rozhraní, využívat generika a další pokročilé funkce, které přispívají k lepší organizaci a škálovatelnosti kódu. Díky tomu je vývoj strukturovanější a méně náchylný k chybám. Kompatibilita TypeScriptu s JavaScriptem a jeho široká podpora v moderních vývojových nástrojích navíc umožňují snadnou integraci s ostatními vybranými technologiemi pro projekt.

2.4 Adresářová struktura

- `app/` – Obsahuje definice rozvržení a jednotlivých stránek aplikace pro Next.js app router.
- `components/` – Složka pro React komponenty, které se používají napříč celou aplikací.
- `collections/` – Obsahuje konfigurace kolekcí pro Payload CMS.
  - Kolekce jsou skupiny záznamů se společným schématem, které slouží k organizaci opakujících se dat v aplikaci (např. uživatelé, stránky, aktuality).
- `globals/` – Obsahuje konfiguraci globals v Payload CMS.
  - Globals jsou globální objekty, které mohou uchovávat sdílená data v aplikaci.
- `fields/` – Obsahuje znovupoužitelná pole pro Payload CMS.
  - Jednotlivé typy polí mohou být sdíleny mezi více kolekcemi, což usnadňuje správu a standardizaci datových struktur.
- `migrations/` – Obsahuje vygenerované migrace pro správu změn v databázi.

  - Migrace jsou využívány v produkčním prostředí k aktualizaci databázového schématu bez ztráty dat.
- `lib/` – Obsahuje pomocné utility a funkce, které se využívají v různých částech aplikace.

3. Technická dokumentace

Živou webovou stránku naleznete na adrese `https://gjk-gamma.vercel.app/`.

3.1 Instrukce k spuštění

3.1.1 Předpoklady

- Máte nainstalováno `npm` / `pnpm` / `yarn`
- Máte zprovozněnou vlastní PostgreSQL databázi

3.1.2 Příprava `.env` souboru

1. Zkopírujte ukázkový soubor:

```
cp .env.example .env
```

2. Vyplňte hodnoty v `.env` tak, aby odpovídaly vašemu prostředí.

3.1.3 Spuštění aplikace

1. Stažení závislostí:

```
npm install  # nebo
pnpm  # nebo
yarn
```

2. Sestavení aplikace:

```
npm run ci  # nebo
pnpm run ci  # nebo
yarn ci
```

3. Spuštění aplikace:

```
npm run start  # nebo
pnpm start  # nebo
yarn start
```

Webové stránky by měly být dostupné na `localhost:3000`, pokud není v záznamech uvedeno jinak.

3.2 Přístup k admin rozhraní

Po spuštění webu je administrátorské rozhraní k dispozici v podadresáři `/admin`, např. `localhost:3000/admin`. Pokud ještě nebyli vytvořeni žádní uživatelé, objeví se panel pro vytvoření nového uživatele, jinak se objeví přihlašovací panel.

Závěr

Cílem této práce bylo vytvořit webovou stránku, která by mohla sloužit jako alternativa pro aktuální školní web GJK. Po dokončení projektu mohu říci, že finální webová stránka splňuje základní požadavky zadání maturitní práce i specifikace stanovené panem ředitelem. Nicméně jsem si vědom některých oblastí, které by bylo možné ještě vylepšit. Největší slabinou vidím v designu stránky, který je v současné podobě poněkud strohý. Také některé funkce navíc, jako například galerie fotek, komplexnější layout stránek nebo upravitelná domovská stránka, které jsem měl v plánu implementovat, nebyly dokončeny kvůli nedostatku času. Tyto aspekty bych rád vylepšil v budoucnosti, pokud bych na projektu pokračoval.

Během práce na tomto projektu jsem se nejen prohloubil své zkušenosti s technologiemi jako React, Next.js a Typescript, ale také jsem se naučil, jak důležitá je kvalitní komunikace a pečlivé plánování při vývoji složitějších webových aplikací, zejména když jsou určeny pro instituci. Naučil jsem se také pracovat s Payload CMS, což mi umožnilo lépe pochopit principy headless CMS systémů a jejich využití pro správu obsahu na webových stránkách. Navíc jsem získal praktické zkušenosti s migracemi SQL databází v produkčním prostředí, což je dovednost, kterou jsem dříve neměl možnost dostatečně prozkoumat.

Tento projekt mi pomohl nejen prohloubit mé technické dovednosti, ale také mi poskytl cenné zkušenosti v oblasti projektového řízení a práce s klientem. Věřím, že zkušenosti získané při práci na tomto projektu mi pomohou v budoucnosti.

Seznam použité literatury

[Mik20] James Mikrut. *Payload vs. Directus vs. Strapi — GraphQL Performance Benchmarks*. Zář. 2020. URL: https://payloadcms.com/blog/performance-benchmarks (cit. 02. 03. 2025).

Seznam obrázků

Seznam tabulek