DELTA – Střední škola informatiky a ekonomie, s.r.o.

Ke Kamenci 151, Pardubice

Dlouhodobý monitor televizního programu

Příjmení, jméno	**Daniel Marek**
Třída	**4.B**
Studijní obor	**Informační technologie 18-20-M/01**
Školní rok	**2024/2025**

Zadání maturitního projektu z informatických předmětů

Jméno a příjmení: *Daniel Marek*

Pro školní rok: *2024/2025*

Třída: *4. B*

Obor: *Informační technologie 18-20-M/01*

Téma práce: *Nástroj pro sledování TV programu*

Vedoucí práce: *RNDr. Jan Koupil, Ph.D.*

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:

Cíl projektu: Cílem tohoto projektu je vytvořit webovou aplikaci, která bude sloužit jako dlouhodobý monitor televizního programu. Aplikace bude agregovat televizní programy z různých online zdrojů a umožní uživatelům zadat filmy či seriály, které chtějí sledovat. Následně budou uživatelé dostávat upozornění, kdy se jejich vybraný obsah bude vysílat.

Specifikace projektu:

1. **Analýza a návrh:**
   - Analýza dostupných zdrojů televizních programů online.
   - Definování požadavků na systém a návrh uživatelského rozhraní.
   - Návrh architektury aplikace pro snadné přidávání zdrojů TV programu.
   - Návrh databázové struktury pro ukládání preferencí a programů.
2. **Vývoj uživatelského rozhraní:**
   - Intuitivní webové rozhraní pro zadání filmů a seriálů.
   - Přehledné zobrazení informací o televizním programu.
3. **Agregace dat (scraping):**
   - Skripty pro stahování televizních programů.
   - Pravidelná aktualizace těchto dat.
4. **Upozornění a notifikace:**

- Systém pro zasílání upozornění (e-mail, webová aplikace).
- Možnost nastavení frekvence a formátu upozornění.

5. **Vyhledávání a filtrování:**
   - Vyhledávací a filtrovací funkce pro oblíbené pořady.
   - Parametry pro vyhledávání (žánr, čas vysílání).
6. **Dobře navržená architektura aplikace:**
   - Modulární architektura s více vrstvami.
   - Oddělené vrstvy (logická, datová, prezentační).

Požadované výstupy:

- Funkční webová aplikace.
- Uživatelská a vývojářská dokumentace.
- Prezentace a demonstrační video.
- Systém pro agregaci a pravidelnou aktualizaci dat.
- Implementovaný systém notifikací.

Hodnocení:

- Kvalita a funkčnost uživatelského rozhraní.
- Schopnost správně agregovat a aktualizovat data.
- Implementace a funkčnost systému notifikací.
- Úroveň dokumentace a prezentace.
- Kreativita a inovativnost řešení.

Stručný časový harmonogram (s daty a konkretizovanými úkoly):

- **Září:** Analýza problému, návrh UI a architektury, návrh databáze.
- **Říjen:** Implementace základního webového rozhraní.
- **Listopad:** Vývoj skriptů pro scraping a aktualizaci dat.
- **Prosinec:** Návrh a implementace notifikačního systému.
- **Leden:** Implementace vyhledávacích funkcí.
- **Únor – březen:** Práce na dokumentaci.

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

V Pardubicích dne _____ _____

Daniel Marek

Poděkování

Děkuji vedoucímu mého projektu RNDr. Janu Koupilovi, Ph.D. za vedení mého projektu a trpělivost se mnou.

Anotace

Cílem projektu je vytvoření plně funkčního a uživatelsky přívětivého systému pro sledování a spravování televizního vysílání. Webová aplikace umožní uživatelům jednoduše sledovat televizní program, nastavovat notifikace k vybraným pořadům a také k pořadům, které se teprve plánují a dosud nebyly oficiálně oznámeny. Aplikace je určena především pro diváky, kteří nechtějí zmeškat své oblíbené pořady, ale zároveň nechtějí trávit čas neustálým kontrolováním televizního programu. Systém tak nabízí chytré a efektivní řešení, které uživatelům ulehčí orientaci ve vysílacím programu klasické televize i v době rozmachu streamovacích služeb.

Klíčová slova

React, Webová aplikace, Next JS, TypeScrpit, Rust, Cron, JavaScript, PostgreSQL, Supabase, Televize

Obsah

1 Úvod **10**

2 Technologie **10**

2.1 TypeScript 10

2.2 React 10

2.3 NextJS 11

2.4 Rust 12

2.5 Cron 13

2.6 PostgreSQL 16

2.6.1 Hlavní vlastnosti PostgreSQL 17

2.7 Supabase 17

3 Jiná řešení **18**

3.1 mujtvprogram.cz 18

3.2 tvtime.com 18

4 Architektura **19**

4.1 Next.JS Aplikace 19

4.2 Supabase Auth 19

4.3 Rust Endpoints 20

4.3.1 Ukládání dat 21

4.4 Web Scraper 22

4.4.1 Princip fungování 22

4.4.2 Ukládání dat 22

4.4.3 Automatizace 22

5 Webová aplikace **23**

5.1 Registrace 23

5.2 Přihlášení 24

5.3 Hlavní stránka 25

5.4 Stránka s notifikacema 27

5.5 Admin page 29

6 Závěr **30**

1 Úvod

V době streamovacích služeb, které nabízejí tisíce filmů a seriálů online, se může zdát, že klasické televizní vysílání ustupuje do pozadí. Přesto se však stále najde mnoho pořadů a filmů, které jsou dostupné výhradně v televizním programu a nelze je jednoduše dohledat na internetu. Pro diváky, kteří nechtějí přijít o své oblíbené pořady, ale zároveň nechtějí neustále kontrolovat televizní program, je potřeba chytré aplikace, která tuto činnost usnadní.

Cílem této aplikace je vytvořit systém, který bude sledovat televizní programy různých kanálů a v případě, že se začne vysílat uživatelem sledovaný pořad, automaticky odešle notifikaci. Uživatel tak získá jednoduchý nástroj, díky kterému nezmešká žádné vysílání svého oblíbeného filmu, seriálu nebo pořadu.

2 Technologie

V této části se podíváme na technologie této aplikace jako jsou frameworky, jazyky a databáze.

2.1 TypeScript

TypeScript je open-source programovací jazyk vyvinutý společností Microsoft, který rozšiřuje JavaScript o statické typování. Jeho cílem je zlepšit udržovatelnost, čitelnost a škálovatelnost kódu, a díky kompilaci odhalovat chyby již v průběhu vývoje, což snižuje pravděpodobnost chyb při běhu aplikace. TypeScript je plně kompatibilní s existujícím JavaScriptem, což usnadňuje jeho integraci do stávajících projektů. Díky svému přísnému typovému systému výrazně usnadňuje vývoj a ladění aplikací. V roce 2025 se TypeScript stal standardem pro vývoj velkých a komplexních aplikací, přičemž jeho popularita stále roste [1].

2.2 React

React je open-source knihovna pro tvorbu uživatelských rozhraní, kterou vyvinula společnost Meta (dříve Facebook). Patří mezi nejpopulárnější technologie pro vývoj moderních webových aplikací. Je založen na **komponentách**, které jsou znovupoužitelné a snadno kombinovatelné, což usnadňuje správu rozsáhlých aplikací.

React využívá **virtuální DOM**, který optimalizuje aktualizaci uživatelského rozhraní a zajišťuje vysoký výkon aplikace. Díky **React Hooks** (např. `useState`, `useEffect`, `useContext`) lze efektivně spravovat stav aplikace a reagovat na změny v průběhu životního cyklu komponenty bez nutnosti třídních komponent.

Pro vývoj v TypeScriptu se využívá **TSX**, což je rozšíření JSX umožňující typovou kontrolu. Výhody TSX v Reactu zahrnují:

- **Statickou typovou kontrolu** – umožňuje odhalit chyby už při vývoji.
- **Lepší dokumentaci kódu** – díky typům jsou komponenty lépe čitelné a snadněji udržovatelné.
- **Bezpečnější manipulaci s props a stavem** – pomocí rozhraní (`interface`) nebo typů (`type`).

React podporuje komunikaci mezi komponenty prostřednictvím **props** a **callback funkcí**. Pro správu složitějšího stavu lze využít:

- **React Context API**
- **Redux**
- **Zustand**
- **Recoil**

React a TypeScript jsou ideální kombinací pro robustní a škálovatelné webové aplikace, a s moderními nástroji jako **Next.js** umožňují i **server-side rendering (SSR)** a optimalizované načítání komponent [2].

2.3 NextJS

Next.js je moderní open-source framework pro vývoj webových aplikací a statických webů založený na Reactu. Byl vyvinut společností **Vercel** a nabízí řadu funkcí, které usnadňují vývoj a optimalizaci React aplikací.

Mezi hlavní vlastnosti Next.js patří:

- **Server-side rendering (SSR)** – umožňuje vykreslování obsahu na serveru pro rychlejší načítání.

- **Statické generování stránek (SSG)** – generování statických HTML souborů pro lepší výkon a SEO.
- **Automatická optimalizace kódu** – minimalizuje velikost bundle a zvyšuje efektivitu aplikace.
- **Incremental Static Regeneration (ISR)** – umožňuje aktualizovat statický obsah bez nutnosti celého rebuildu.

Díky těmto funkcím umožňuje Next.js vytvářet rychlé a výkonné aplikace. Díky své flexibilitě a širokým možnostem získává stále více popularity a je vhodný pro různé typy projektů – od malých webů až po rozsáhlé firemní aplikace [3].

2.4 Rust

Rust je moderní systémový programovací jazyk, který byl původně vyvinut ve společnosti Mozilla Research [4]. Zaměřuje se na **bezpečnost, výkon a souběžnost**, což jej činí ideální volbou pro vývoj nízkoúrovňových aplikací, jako jsou operační systémy, vestavěné systémy nebo webové servery.

Rust je specifický svou **přísnou správou paměti** bez nutnosti garbage collectoru. Místo toho využívá systém vlastnictví (*ownership*), půjčování (*borrowing*) a doby života (*lifetimes*), které zajišťují bezpečnost paměti během kompilace.

Hlavní výhody Rustu zahrnují:

- **Bezpečnost paměti** – minimalizuje chyby jako *null pointer dereference* nebo *buffer overflow*.
- **Vysoký výkon** – optimalizovaný kód srovnatelný s jazykem C nebo C++.
- **Bezpečná souběžnost** – Rust umožňuje efektivní práci s více vlákny bez rizika datových závodů.
- **Skvělý ekosystém** – bohatá knihovna `std`, správce balíčků Cargo a rozsáhlá komunita.

Rust nachází uplatnění v mnoha oblastech, včetně vývoje systémových aplikací, blockchainových technologií, síťových služeb a webových aplikací. Mezi známé projekty využívající Rust patří například Firefox Servo, Dropbox, Cloudflare nebo Discord.

2.5 Cron

Cron je systémový plánovač úloh v Unixových operačních systémech [5]. Umožňuje automatické spouštění skriptů a příkazů v předem definovaných intervalech bez nutnosti ručního zásahu uživatele. Typickým využitím cronu jsou úlohy jako zálohování dat, aktualizace systémových balíčků, pravidelné reporty nebo mazání nepotřebných souborů.

Cron poskytuje administrátorům a uživatelům možnost plánovat úlohy opakovaně nebo jednorázově, což zvyšuje efektivitu správy systému a minimalizuje manuální práci.

Crontab – Konfigurace úloh

Úlohy pro cron jsou definovány v souboru `crontab`, kde každý řádek představuje jednu naplánovanou úlohu ve formátu:

```
* * * * * command_to_execute
```

Každá hvězdička reprezentuje specifickou časovou jednotku[7]:

Pozice	Význam	Hodnoty
1.	Minuty	0–59
2.	Hodiny	0–23
3.	Dny v měsíci	1–31
4.	Měsíce	1–12
5.	Dny v týdnu	0–7 (0 a 7 = neděle)

Tabulka 1: Struktura crontabu [6]

Každá z těchto časových jednotek může být specifikována několika způsoby [9]:

- **Jednotlivá hodnota:** např. `5` znamená konkrétní časovou jednotku (5. minuta).
- **Seznam hodnot:** např. `0,15,30,45` spustí příkaz každých 15 minut.
- **Rozsah:** např. `9-17` znamená každou hodinu mezi 9. a 17. hodinou.
- **Kroková hodnota:** např. `*/10` znamená každých 10 jednotek (např. každých 10 minut).

Příklady použití

Níže jsou uvedeny příklady pro lepší pochopení konfigurace úloh v crontabu:

```
# Každý den o~půlnoci
0 0 * * * /usr/bin/daily_backup.sh

# Každé pondělí v~6:30
30 6 * * 1 /home/user/weekly_report.sh

# Každých 10 minut
*/10 * * * * /home/user/monitoring_script.sh

# Jednou za měsíc, první den měsíce v~5:00 ráno
0 5 1 * * /home/user/monthly_cleanup.sh

# Pouze v~pracovní dny v~17:00
0 17 * * 1-5 /home/user/workday_notification.sh
```

Pokročilá konfigurace

Cron umožňuje využívat speciální předdefinované zkratky pro běžné intervaly:

Zkratka	**Význam**
`@reboot`	Spustí se po startu systému
`@hourly`	Každou hodinu (ekvivalent `0 * * * *`)
`@daily`	Denně o půlnoci (ekvivalent `0 0 * * *`)
`@weekly`	Každý týden v neděli o půlnoci (ekvivalent `0 0 * * 0`)
`@monthly`	Každý měsíc první den o půlnoci (ekvivalent `0 0 1 * *`)
`@yearly`	Jednou ročně 1. ledna o půlnoci (ekvivalent `0 0 1 1 *`)

Tabulka 2: Speciální zkratky v crontabu

Příklad použití speciálních zkratek:

```
@reboot /home/user/startup_script.sh
@daily /home/user/daily_maintenance.sh
```

Správa crontabu

Crontab lze upravovat příkazem:

```
crontab -e
```

Další užitečné příkazy:

- `crontab -l` – zobrazí aktuální úlohy cronu.
- `crontab -r` – smaže celý crontab uživatele (používejte s opatrností).

Umístění crontab souborů

V systému existují specifické adresáře, kam se ukládají systémové cron soubory:

- `/etc/crontab` – systémové úlohy a skripty, typicky spravované administrátorem.
- `/etc/cron.d/` – jednotlivé soubory pro specifické aplikace.
- `/etc/cron.hourly`, `cron.daily`, `cron.weekly`, `cron.monthly` – adresáře pro skripty, které se spouští pravidelně v odpovídajících intervalech.

Diagnostika a řešení problémů

Pokud cron nefunguje dle očekávání, doporučuje se provést následující kontroly:

1. Ověřte syntax crontabu (např. příkazem `crontab -l`).
2. Zkontrolujte práva spuštění skriptu (`chmod +x script.sh`).
3. Použijte absolutní cesty, protože cron má omezené prostředí (`/usr/bin/bash`, místo pouze `bash`).
4. Ověřte logy cronu (`/var/log/syslog`, `/var/log/cron`).

Příklad kontroly logu pro cron:

```
grep CRON /var/log/syslog
```

To pomůže identifikovat případné chyby a zjistit, proč se úlohy nespouštějí.

Zabezpečení cronu

Cron je výkonný nástroj a může být bezpečnostním rizikem, pokud je špatně nakonfigurován. Pro omezení přístupu k cronu existují soubory:

- `/etc/cron.allow` – seznam uživatelů, kteří mohou cron používat.
- `/etc/cron.deny` – seznam uživatelů, kteří cron používat nemohou.

Pokud soubor `cron.allow` existuje, jen uživatelé v něm uvedení mohou cron využívat. V opačném případě platí omezení dle `cron.deny`.

Správná konfigurace a důkladná kontrola cron úloh je klíčová pro stabilní a bezpečný provoz systémů založených na Unixu a Linuxu.

Příklady použití

Frekvence	**Syntaxe**	**Příkaz**
Každý den ve 2:30 ráno	`30 2 * * *`	`/home/user/backup.sh`
Každé pondělí ve 12:00	`0 12 * * 1`	`/home/user/report.sh`
Každých 5 minut	`*/5 * * * *`	`/home/user/log_cleanup.sh`

Tabulka 3: Příklady použití cronu

Správa Crontabu

Akce	**Příkaz**
Zobrazení aktuálních úloh	`crontab -l`
Editace crontabu	`crontab -e`
Smazání všech úloh	`crontab -r`

Tabulka 4: Správa crontabu

2.6 PostgreSQL

PostgreSQL je výkonný open-source relační databázový systém [10]. Byl vyvinut jako pokročilá alternativa k tradičním SQL databázím a nabízí širokou škálu funkcí, které jej činí jedním z nejpoužívanějších databázových řešení pro moderní aplikace.

2.6.1 Hlavní vlastnosti PostgreSQL

- **Podpora ACID** – Atomicita, Konzistence, Izolace a Trvanlivost zajišťují spolehlivost databázových operací.
- **Pokročilé dotazování** – Podpora složitých SQL dotazů, včetně CTE, JSONB, full-textového vyhledávání a analytických funkcí.
- **Škálovatelnost** – Možnost replikace, sharding a podpora velkých objemů dat.
- **Rozšířitelnost** – Možnost přidávání vlastních funkcí a modulů, jako jsou PL/pgSQL, Python nebo C.

PostgreSQL je využíváno v mnoha oblastech včetně finančních systémů, analytických platforem, datových skladů a webových aplikací [10].

2.7 Supabase

Supabase je moderní open-source alternativa k Firebase, postavená na PostgreSQL [11]. Poskytuje backend-as-a-service řešení, které umožňuje vývojářům snadno integrovat databázi, autentizaci, serverless funkce a úložiště souborů do jejich aplikací.

Hlavní vlastnosti Supabase

- **PostgreSQL databáze** – Plná podpora PostgreSQL včetně pokročilých dotazovacích funkcí.
- **Autentizace a správa uživatelů** – Integrovaná autentizace přes e-mail, OAuth a magic links.
- **Realtime API** – Automatická aktualizace klientů při změně dat díky využití **PostgreSQL listen/notify.**
- **Cloudové úložiště** – Možnost ukládání a správy souborů přímo v Supabase.
- **Serverless funkce** – Možnost vytváření vlastních backendových funkcí.

Supabase je ideální volbou pro vývojáře, kteří hledají open-source backend s bohatými funkcemi, aniž by se museli spoléhat na proprietární řešení jako Firebase.

[11]

3 Jiná řešení

Existuje celá řada aplikací na sledování TV programů, s různými funkcemi a postavených na různých technologiích. Zde si je ukážeme, abysme měli přehled o konkurenčních řešeních.

3.1 mujtvprogram.cz

Jako první se podíváme na aplikaci mujtvprogram, která mi sloužila jako inspirace pro frontend.

Mujtvprogram je jedna z nejpopulárnějších aplikací na zaznamenávání upozorňování o tv pořadech.

Narozdíl od mého programu, mujtvprogram nelze nastavit tak aby upozorňoval na pořady, které ještě nejsou officiálně oznámeny ve vysílání, lze nastavit upozrnění jen na pořady, které se budou vysílat.

Celý program je napsán v php.

3.2 tvtime.com

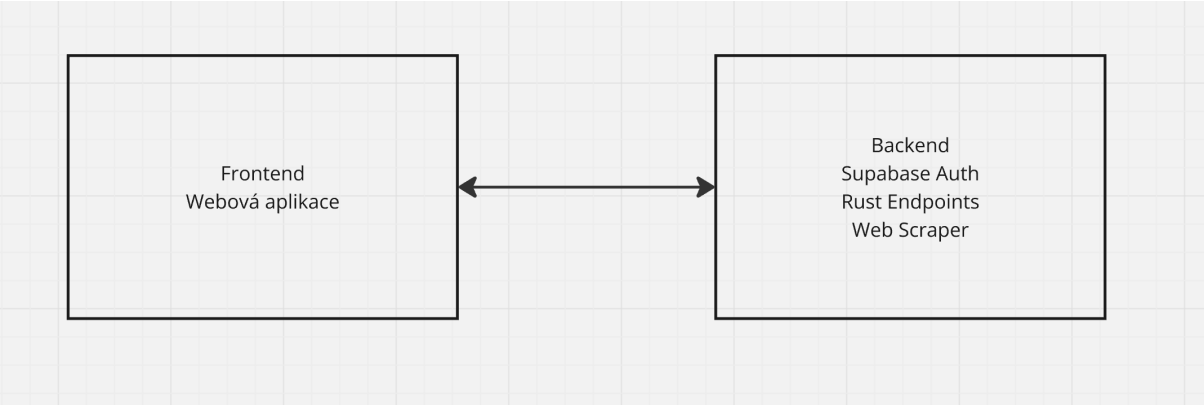
TV Time je jedna z nejpopulárnějších aplikací pro sledování a organizaci oblíbených seriálů a filmů.

Na rozdíl od mého programu, TV Time neumožňuje upozorňování na pořady, které ještě nebyly oficiálně oznámeny, ale pouze na epizody a filmy, které mají stanovené datum vydání.

Celý program je dostupný jako mobilní aplikace pro Android a iOS a je zaměřen na komunitní interakci a přizpůsobené doporučení podle sledovaných pořadů.

4 Architektura

Architekturu aplikace tvoří 4 prvky, Next.JS (webová aplikace), Supabase Auth, Rust Endpoints a Web Scrapper 1.



Obrázek 1: Architektura

4.1 Next.JS Aplikace

Next.JS je nadstavbou reactu a poskytuje prvky pro snadnější práci.

Ve složce „/app“ se nachází všechny stránky, na které se můžeme v aplikaci dostat, výjimkou jsou akorát stránky, které mají na začátku názvu podtržítko.

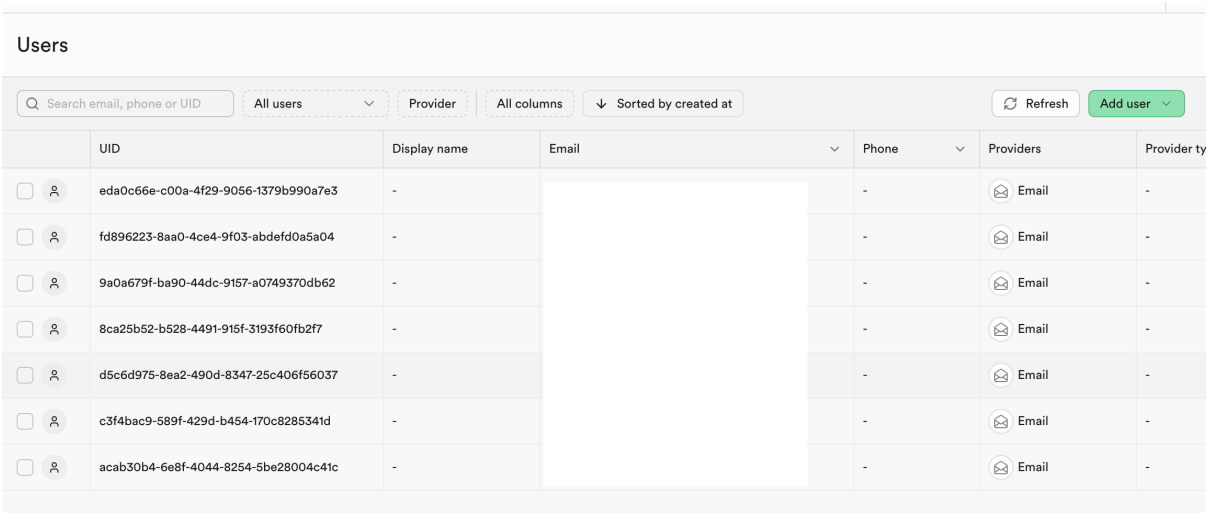
Složka „/utils“ obsahuje pomocné funkce, utility a další moduly, které se používají napříč celou aplikací. Tyto soubory slouží ke zjednodušení kódu a usnadnění opakovaně používaných operací.

4.2 Supabase Auth

Pro autentizaci uživatelů zde používáme Supabase Auth, což je moderní a flexibilní autentizační služba postavená na PostgreSQL. Umožňuje snadné přihlašování a registraci uživatelů s podporou více metod ověřování.

V této aplikaci využíváme login/registraci pouze přes e-mail a heslo, ale Supabase Auth také podporuje přihlášení pomocí OAuth providerů, jako jsou Google, Facebook, GitHub a další. Navíc umožňuje ověřování pomocí OTP kódů (SMS/magic link) nebo SAML a OpenID Connect pro podnikové aplikace.

Údaje o uživatelých se ukládají do tabulky auth.users 2



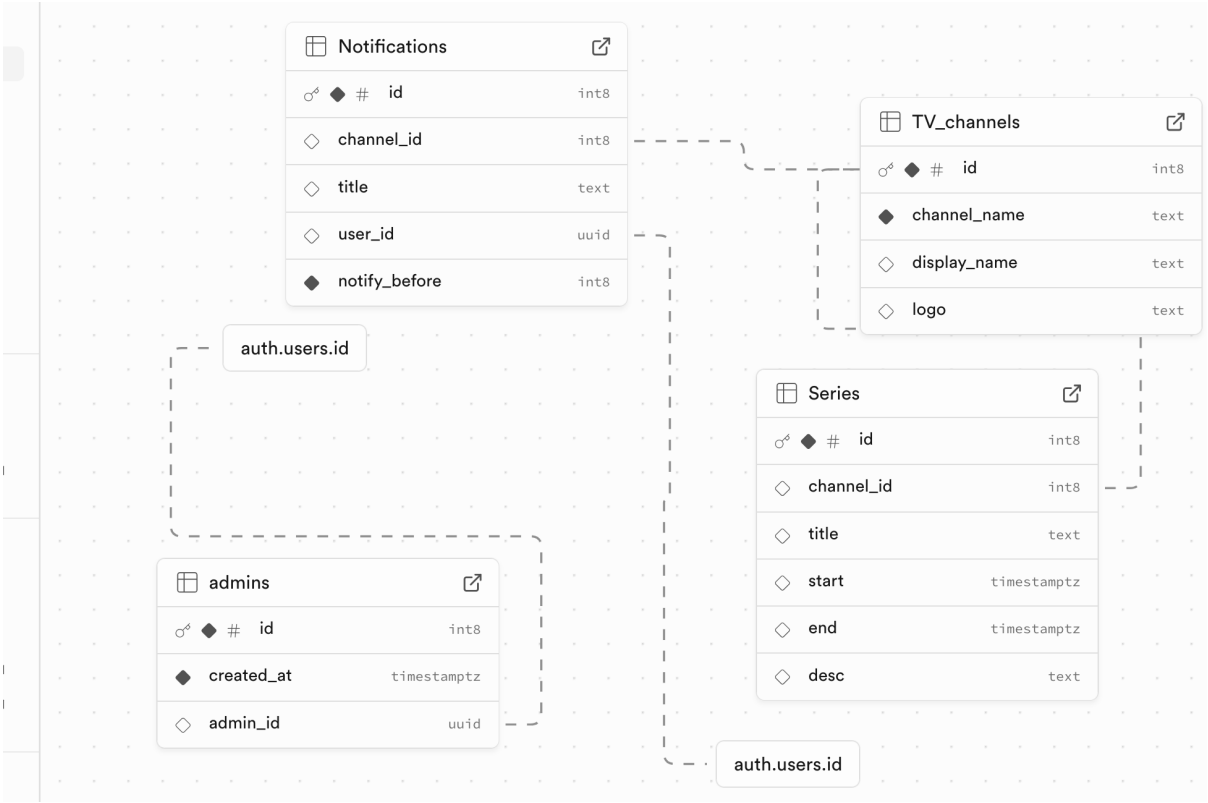
Obrázek 2: Supabase.auth

V tabulce se nachází

- **id** – unikátní identifikátor uživatele (UUID)
- **email** – e-mailová adresa uživatele (hlavní identifier)
- **created_at** – datum a čas vytvoření účtu
- **last_sign_in_at** – poslední přihlášení uživatele
- **role** – role uživatele (pokud je definována)
- **app_metadata** – metadata spojená s autentizací, např. poskytovatel přihlášení
- **user_metadata** – dodatečná uživatelská data (pokud jsou nastavena)

4.3 Rust Endpoints

Pro backend v této aplikaci využíváme Rust. Komunikace mezi klientem a serverem probíhá pomocí HTTP požadavků, které jsou odesílány na příslušné API endpointy. K odesílání požadavků používáme knihovnu reqwest, která umožňuje snadnou integraci HTTP klienta do aplikace.



Obrázek 3: Struktura databáze

4.3.1 Ukládání dat

Jako databázi používáme SQL databázi s názvem Supabase, která je postavena na amazon SDK.

Na obrázku č. 3 vidíme strukturu naší databáze v Supabase. Základem jsou tabulky, které obsahují jednotlivé entity s definovanými poli, podobně jako v klasických relačních databázích. V těchto polích můžeme definovat různé datové typy, například text, integer, boolean, timestamp nebo uuid. V našem případě používáme například timestamp pro ukládání časových údajů, což je pro naši aplikaci ideální řešení.

V naší databázi využíváme několik tabulek, mezi nimiž jsou například:

- **tv_channels** – obsahuje informace o televizních kanálech
- **series** – ukládá informace o jednotlivých seriálech a jejich časech vysílání
- **notifications** – uchovává notifikace pro uživatele spojené s kanály nebo seriály

Tabulky jsou mezi sebou propojené pomocí cizích klíčů, což nám umožňuje efektivní dotazování a správu dat.

Každý dokument v tabulkách je reprezentován unikátním identifikátorem, který je generován buď automaticky Supabase, nebo specifikován uživatelem. Díky PostgreSQL backendu máme k dispozici pokročilé dotazovací možnosti a možnosti škálování pro budoucí rozšíření aplikace.

4.4 Web Scraper

Pro automatické získávání dat využíváme web scraper, který běží každý den a aktualizuje databázi s nejnovějšími informacemi o televizních pořadech.

4.4.1 Princip fungování

Každý den v předem stanovený čas se spustí program napsaný v JavaScriptu, který prochází webové stránky poskytující informace o televizním vysílání. Pomocí knihovny **Puppeteer** získává potřebná data, která následně parsuje a ukládá do databáze.

4.4.2 Ukládání dat

Získaná data jsou po zpracování odeslána do **Supabase** databáze, kde jsou ukládána do příslušných tabulek. Například:

- **TV_channels** - uchovává informace o televizních kanálech
- **series** – uchovává detaily o pořadech
- **notifications** – generuje upozornění pro uživatele na základě nově přidaných pořadů

4.4.3 Automatizace

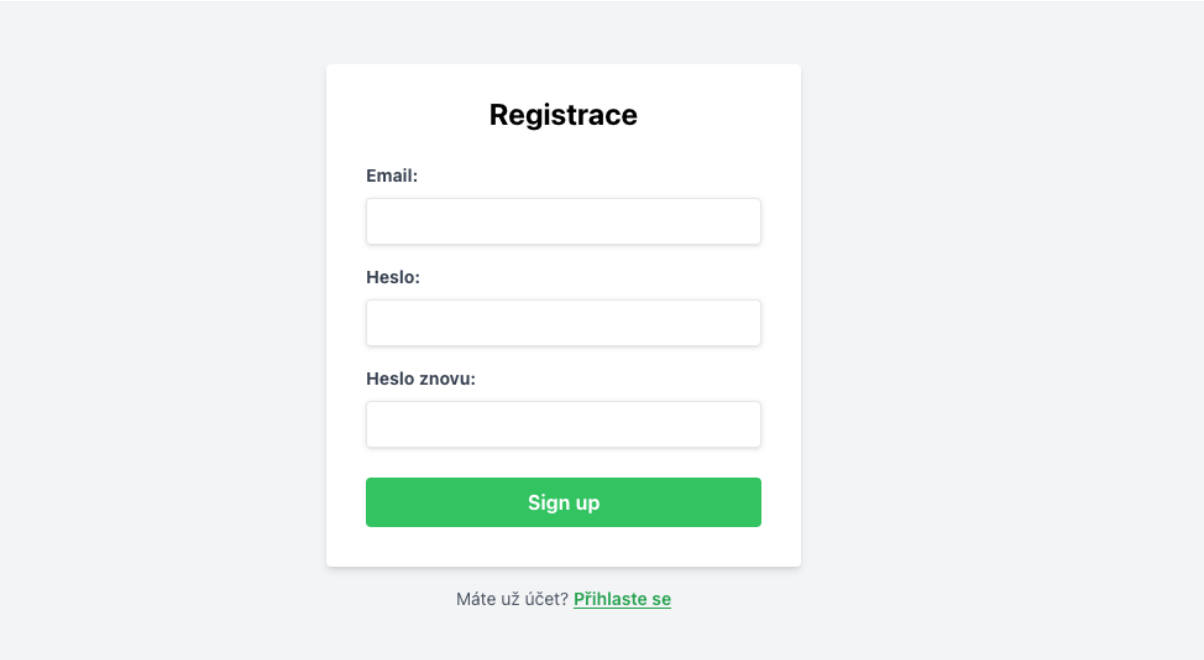
Spuštění scraperu je řízeno pomocí **cron jobu**, která zajistí jeho běh každý den ve stejnou dobu. Tento přístup umožňuje neustále aktualizovat databázi bez manuálního zásahu.

Díky tomuto systému má uživatel vždy přístup k nejnovějším informacím o televizním programu, což je klíčové pro správné fungování aplikace.

5 Webová aplikace

V této části si popíšeme veškeré funkce systému přístupné uživatelům.

5.1 Registrace



Obrázek 4: Obrázek registrace

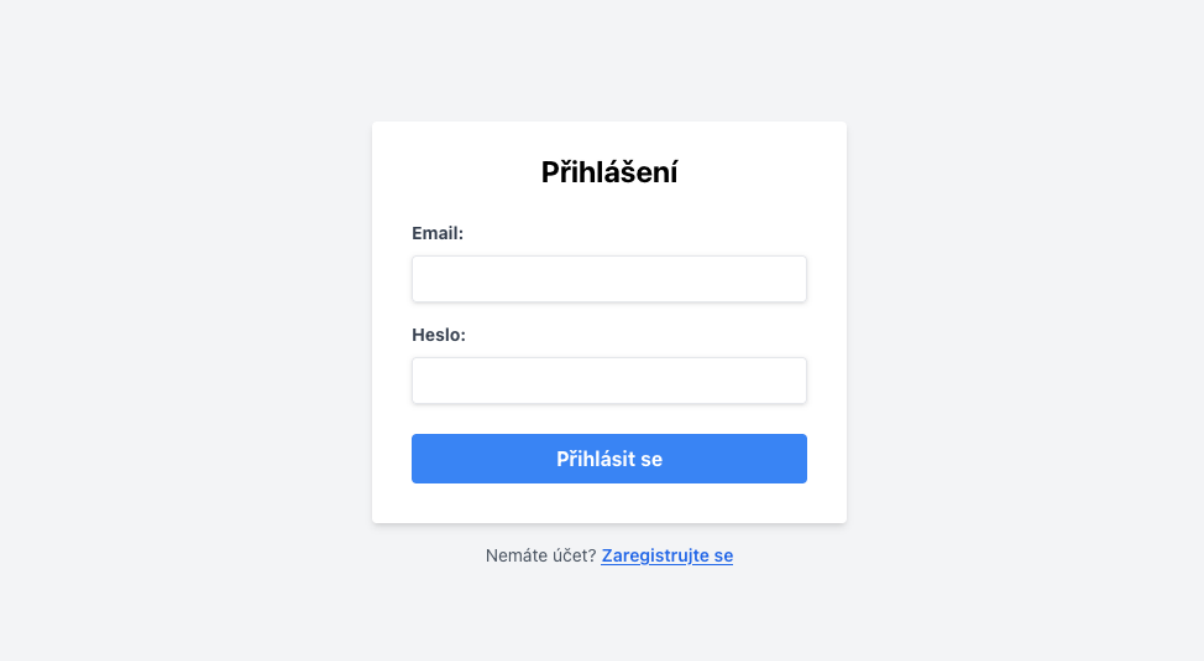
Registrace funguje přes již zmíněnou Supabase auth. UI vidíme na obrázku č. 4. Pro úspěšné fungování aplikace, musí být uživatel zaregistrován a přihlášen. Při registraci se vyplňuje e-mail, heslo a pro ověření heslo ještě jednou. Pro validní registraci, musí být správná forma e-mailu a heslo musí mít minimálně 6 znaků, 1 velké písmeno, jedno malé a číslo. Kód následně ověří, jestli už e- mail není zaregistrován, popřípadě napíše uživateli, že je e-mail už použit. U hesla ověří požadované parametry, jestli se hesla shodují a pokud je nějaký problém, na uživatele vyskočí text s problémem, který můžeme vidět na obrázku č. 5.

```
const errorMessages: Record<string, string> = {
  email_exists: "Tento e-mail je již zaregistrován.",
  invalid_email: "Zadejte platnou e-mailovou adresu.",
  weak_password:
    "Slabé heslo. Zvolte silnější heslo (min. 6 znaků), jedno velké písmeno, jedno malé písmeno a jedno číslo.",
  signup_disabled: "Registrace jsou momentálně vypnuté.",
  unexpected_error: "Nastala neočekávaná chyba. Zkuste to prosím později.",
};
```

Obrázek 5: Chybové hlášky

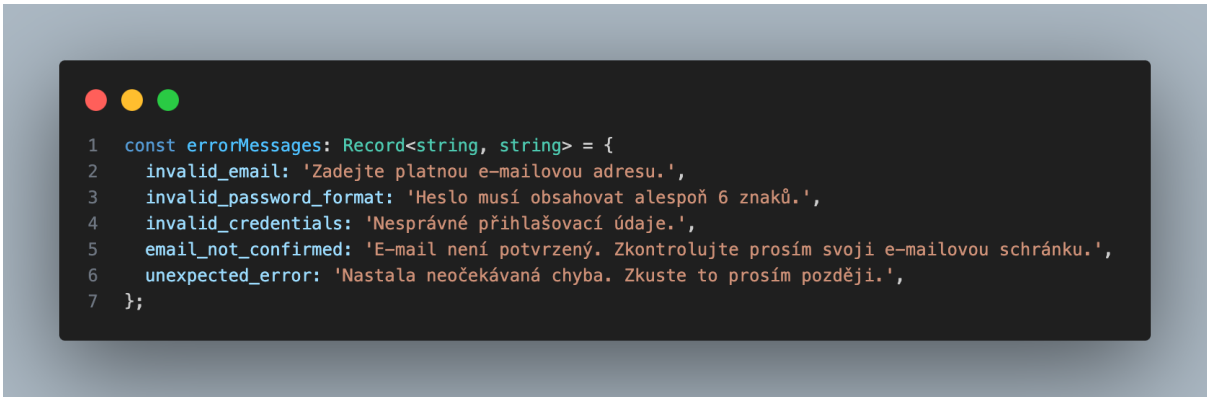
Po úspěšné registraci je uživatel přesměrován na stránku s přihlášením.

5.2 Přihlášení



Obrázek 6: Obrázek přihlášení

Přihlašování funguje stejně jako registrace přes Suapbase Auth. UI vidíme na obrázku č. 6. Uživatel musí být vždy přihlášen, aby mu aplikace fungovala. Při přihlášení se zadává předem zaregistrované heslo a e-mail, které kód ověří, jestli má e-mail správnou formu, je už v databázi a jestli k němu sedí správné heslo, popřípadě vypíše, co je špatně. Chybové hlášky vidíme na obrázku č. 7.

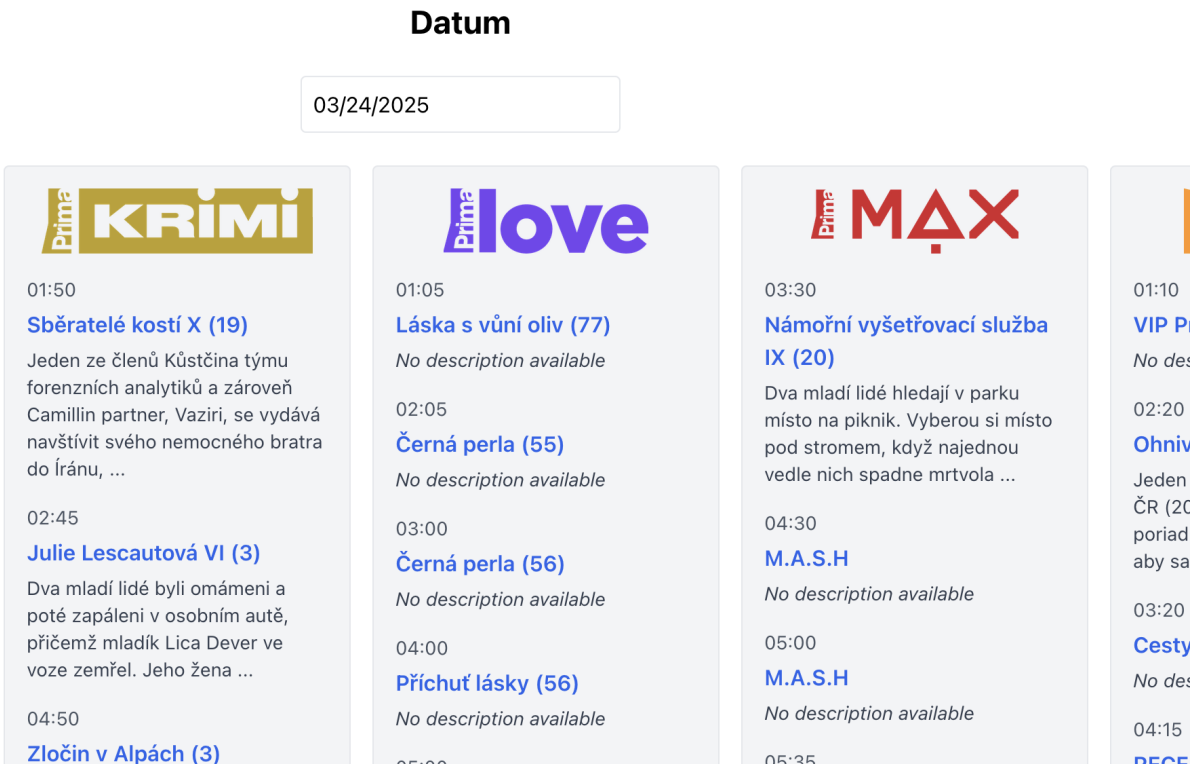


```
const errorMessages: Record<string, string> = {
  invalid_email: 'Zadejte platnou e-mailovou adresu.',
  invalid_password_format: 'Heslo musí obsahovat alespoň 6 znaků.',
  invalid_credentials: 'Nesprávné přihlašovací údaje.',
  email_not_confirmed: 'E-mail není potvrzený. Zkontrolujte prosím svoji e-mailovou schránku.',
  unexpected_error: 'Nastala neočekávaná chyba. Zkuste to prosím později.',
};
```

Obrázek 7: Chybové hlášky

5.3 Hlavní stránka

Na úvodní stránce aplikace je uživateli k dispozici přehled televizních kanálů spolu se seznamem aktuálně vysílaných a připravovaných pořadů. Uživatel si může jednoduše procházet plánované vysílání a vyhledávat pořady, které ho zajímají. Po kliknutí na konkrétní pořad se zobrazí detailní informace, jako je čas vysílání, stručný popis obsahu či herecké obsazení. Součástí funkcionality je také možnost nastavit si upozornění (notifikaci) na vybraný pořad, díky čemuž bude uživatel včas informován před jeho začátkem. Notifikace lze následně spravovat na samostatné stránce, kde je možné jednotlivá upozornění upravovat nebo odstraňovat.



Obrázek 8: Ukázka hlavní stránky s kanálama



Obrázek 9: Ukázka jednoho Pořadu

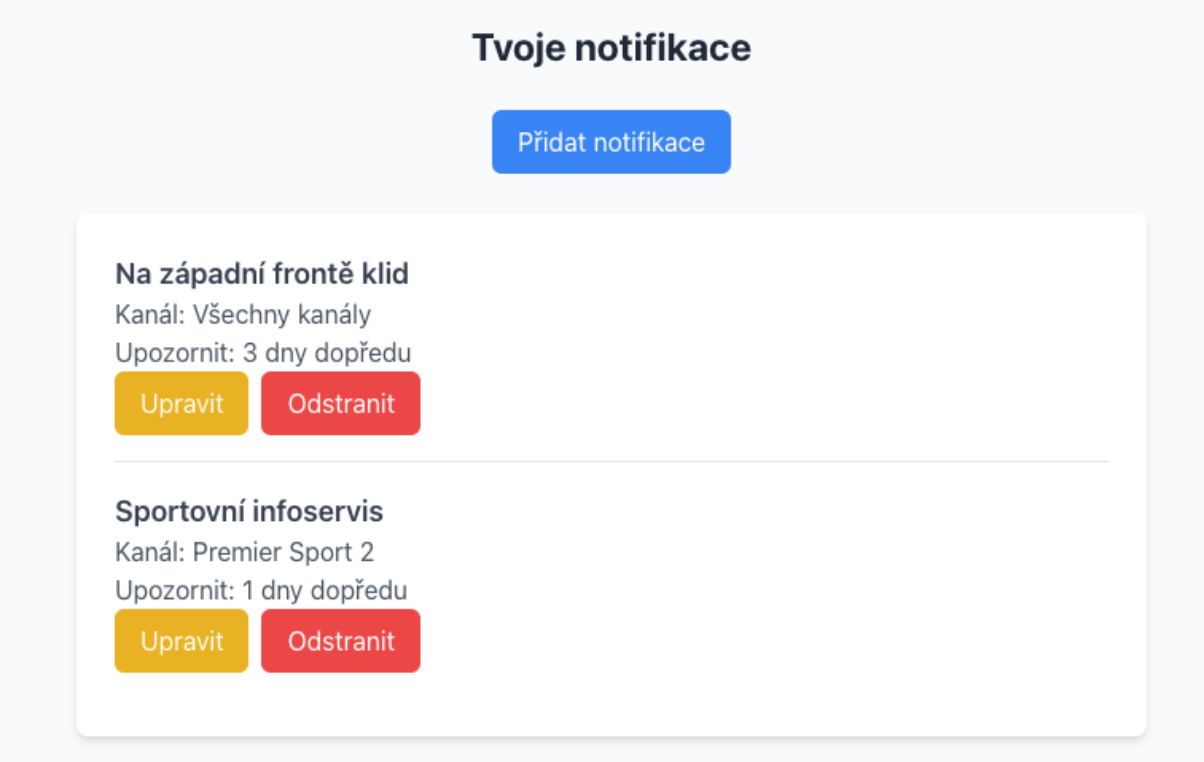
5.4 Stránka s notifikacema

Stránka s notifikacemi slouží uživateli jako přehled všech jeho aktivních i plánovaných upozornění na televizní pořady. Každá notifikace obsahuje základní informace o pořadu, na který je nastavena – například název pořadu, čas vysílání, televizní kanál a případně poznámku, kterou si uživatel přidal.

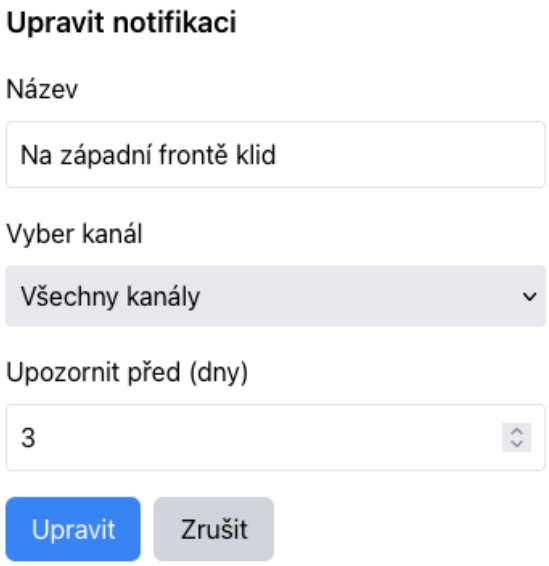
Uživatel má na této stránce možnost s notifikacemi dále pracovat:

- **Zobrazit notifikace** – uživatel vidí všechny své aktuálně nastavené notifikace v přehledném seznamu, seřazeném podle času vysílání nebo podle oblíbenosti.
- **Upravit notifikaci** – kliknutím na konkrétní upozornění může uživatel změnit nastavený čas upozornění (např. 2 dny před začátkem pořadu) nebo upravit další parametry, jako je poznámka či priorita upozornění.
- **Odstranit notifikaci** – pokud uživatel o připomínku nemá zájem, může ji jednoduše smazat a tím zabránit zobrazování upozornění v budoucnu.
- **Přidat novou notifikaci** – přímo ze stránky s notifikacemi může uživatel ručně přidat nové upozornění na libovolný pořad. 16

Díky této funkcionalitě má uživatel plnou kontrolu nad svými upozorněními a může si snadno plánovat sledování televizních pořadů bez obav, že na něco zapomene.



Obrázek 10: Ukázka stránky s notifikacemi



Obrázek 11: Ukázka notifikace

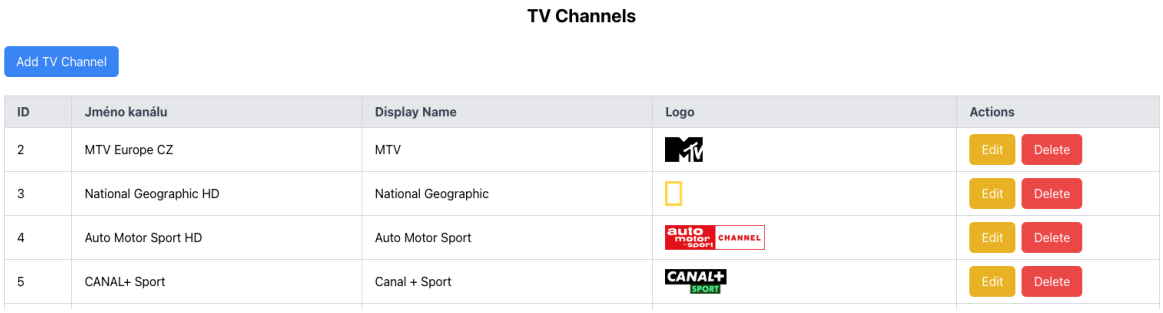
5.5 Admin page

Tato stránka slouží jako administrační rozhraní pro správu kanálů, o kterých mám uložená data. Přístup na stránku je omezen pouze na administrátory, jejichž e-mailové adresy jsou pevně definovány ve statické tabulce `admins`. Tímto způsobem je zajištěna jednoduchá, ale efektivní kontrola přístupu bez nutnosti implementace pokročilé role-based autorizace.

Administrátor má možnost:

- zobrazit seznam všech existujících kanálů,
- přidat nový kanál včetně základních metadat (např. název, popis, URL),
- upravit existující kanály,
- nahrát nebo změnit náhledový obrázek kanálu.

Obrázky jsou ukládány pomocí **Supabase Storage**.



Obrázek 12: Ukázka admin page

6 Závěr

Cílem projektu bylo vytvořit plně funkční a uživatelsky přívětivý systém pro sledování a správu televizního vysílání. Tento cíl se podařilo naplnit – výsledná webová aplikace umožňuje uživatelům snadno procházet televizní program, nastavovat notifikace k oblíbeným pořadům i k pořadům, které teprve budou oznámeny.

Aplikace obsahuje všechny klíčové funkce, jako je registrace, přihlášení, přidávání a odebírání notifikací, a přehled všech dostupných televizních kanálů. Grafické zpracování je minimalistické, což odráží mé dosavadní zkušenosti s návrhem uživatelského rozhraní, nicméně do budoucna bych rád vizuální stránku ještě vylepšil.

Celkově je projekt otestován, plně funkční a připraven k praktickému použití.

Reference

[1] TypeScript in 2025: Enhancing Productivity in Software Development dostupné z: https://www.remoteplatz.com/en/blog/typescript-in-2025--enhancing-productivity-in-soft, [online, cit. 2025-02-18].

[2] React – Oficiální dokumentace, dostupné z: https://react.dev/, [online, cit. 2025-02-18].

[3] Next.js – Oficiální dokumentace, dostupné z: https://nextjs.org/docs, [online, cit. 2025-02-18].

[4] Rust – Oficiální dokumentace, dostupné z: https://www.rust-lang.org/, [online, cit. 2025-02-18].

[5] Paul Vixie. *Cron: UNIX Job Scheduler*. 4.1th ed. FreeBSD Documentation Project, 1994. URL: https://man.freebsd.org/cgi/man.cgi?crontab(5) (navštíveno 19. 3. 2025).

[6] Evi Nemeth, Garth Snyder, Trent R. Hein, Ben Whaley, Dan Mackin. *UNIX and Linux System Administration Handbook*. 5th ed., Addison-Wesley Professional, 2017, s. 785–798. ISBN 978-0134277554.

[7] Mark G. Sobell. *A Practical Guide to Linux Commands, Editors, and Shell Programming*. 4th ed., Addison-Wesley, 2017, s. 497–505. ISBN 978-0134774602.

[8] Michael Kerrisk et al. *Linux Programmer's Manual*. Linux man-pages Project. Dostupné online: https://man7.org/linux/man-pages/man5/crontab.5.html (navštíveno 19. 3. 2025).

[9] Ubuntu Community Documentation. *CronHowTo*. Dostupné online: https://docs.gitlab.com/topics/cron/ (navštíveno 19. 3. 2025).

[10] PostgreSQL – Oficiální dokumentace, dostupné z: https://www.postgresql.org/, [online, cit. 2025-02-18].

[11] Supabase – Oficiální dokumentace, dostupné z: https://supabase.com/docs, [online, cit. 2025-02-18].

Seznam obrázků

1 Architektura 19
2 Supabase.auth 20
3 Struktura databáze 21
4 Obrázek registrace 23
5 Chybové hlášky 24
6 Obrázek přihlášení 24
7 Chybové hlášky 25
8 Ukázka hlavní stránky s kanálama 26
9 Ukázka jednoho Pořadu 26
10 Ukázka stránky s notifikacemi 28
11 Ukázka notifikace 28
12 Ukázka admin page 29