Střední průmyslová škola, Ústí nad Labem, Resslova 5



Playstorm

Dokumentace k ročníkové práci

**Autor:** Martin Černý

**Třída:** 4ITA

**Vedoucí práce:** Bc. Tomáš Ulrich 2025/2026

Prohlášení

Prohlašuji, že jsem ročníkovou práci na téma „Playstorm“ vypracoval samostatně s použitím uvedené literatury a pramenů.

V Ústí nad Labem dne 30.12.2025

……..…………………….

Poděkování

Chtěl bych poděkovat Bc. Tomáši Ulrichovi za vedení mé ročníkové práce, cenné rady a odborný dohled. Děkuji také Bc. Tomáši Ulrichovi za pomoc při gramatické kontrole práce.

Anotace

Sem vložte text anotace.

POKYNY: Anotace obsahuje celkový popis tohoto dokumentu, NIKOLIV popis ročníkové práce jako takové. Popište, z jakých částí se dokumentace skládá, co je v ní představeno, co najdeme v rešerši, co v praktické části, o čem pojednávají jednotlivé části dokumentace. **Minimální délka je 1 odstavec.**

Klíčová slova

Sem vložte seznam klíčových slov.

POKYNY: Napište seznam cca 10 nejpoužívanějších slov, která vystihují téma probírané ve vaší práci. Nejedná se ani o spojky, ani o předložky. Slova pište za sebou, oddělujte je čárkou.

Obsah

[1 Teoretická část 8](#_heading=h.b795qnw7pp3c)

[1.1 Rešerše 8](#_heading=h.6sxb7ql0ebsy)

[1.1.1 Produkt X 8](#_heading=h.6qlr2rinin4g)

[1.1.2 Produkt Y 8](#_heading=h.6b395limx8jt)

[1.2 Technologie 10](#_heading=h.ldbcvkcb4js)

[1.2.1 React & Vite 10](#_heading=)

[1.2.2 Typescript 10](#_heading=)

[1.2.3 Tailwind CSS 10](#_heading=)

[1.2.4 Socket.io 10](#_heading=)

[1.2.5 Vidstack 10](#_heading=)

[1.2.6 Zustand 10](#_heading=)

[1.2.7 NestJS 10](#_heading=)

[1.2.8 Prisma 10](#_heading=h.nhrblb5sfyuu)

[1.2.9 Postgresql 11](#_heading=h.y21eg3z837fh)

[1.2.10 Argon 2 & JWT 11](#_heading=h.8fwdw83y8n90)

[1.2.11 Go (Golang) 11](#_heading=h.4x6xwzvzlo45)

[1.2.12 FFmpeg 11](#_heading=h.iaipfv4adzt6)

[1.2.13 Python 11](#_heading=h.orzhr9oc57lc)

[1.2.14 Docker 11](#_heading=h.8n7t7wq2byow)

[2 Praktická část. 12](#_heading=h.vhyst8tdwbwf)

[2.1 Návrhy 12](#_heading=h.zv8szyqj0t4)

[2.1.1 Návrh Databáze 12](#_heading=h.rkdv5j3qf4k5)

[2.1.2 Návrh backend routing a API 13](#_heading=h.1gqvc93mearc)

[2.2 Produktizace 14](#_heading=h.tkp14y5jrn0i)

[2.3 Popis pro uživatele 14](#_heading=h.yzhoj36ij1w4)

Úvod

Hlavním cílem ročníkové práce je vytvořit webovou aplikaci pro streamování filmů, seriálů a anime. Klíčovou součástí projektu je implementace funkce „Watchparty“ přímo do rozhraní aplikace. Ta umožní skupině lidí sledovat video společně v reálném čase, což znamená, že všichni uvidí totéž a ovládání – jako pauza nebo přetáčení – bude synchronizované pro všechny účastníky najednou.

Téma jsem si vybral, protože mě zajímá systémový design velkých platforem jako Netflix nebo HBO. Chtěl jsem do hloubky pochopit, jak tyto systémy fungují a jakým způsobem pracují s videem ve vysokém rozlišení. Je to pro mě příležitost si v praxi vyzkoušet technologie, které stojí za moderním streamováním.

Od práce očekávám, že se naučím efektivně propojit frontend s backendem a vyřeším technické výzvy spojené se synchronizací uživatelů. Cílem je vytvořit plynulou a funkční aplikaci, která lidem nabídne kvalitní zážitek ze společného sledování.

# Teoretická část

## Rešerše

### Netflix

Globální streamovací platforma zaměřená na online přehrávání filmů a seriálů s důrazem na jednoduché ovládání a personalizaci obsahu. Inspirací pro aplikaci PlayStorm je především přehledné uživatelské rozhraní, které klade obsah do popředí, a princip doporučování na základě chování uživatele. Důležitým prvkem je také stabilní a plynulé streamování, které zajišťuje kvalitní uživatelský zážitek.



Obrázek 1 - Netflix Logo [1]

### Wacth2Gether

Online služba umožňující více uživatelům sledovat video obsah společně v reálném čase. Hlavní inspirací pro aplikaci PlayStorm je koncept synchronizovaného přehrávání a sdíleného zážitku na dálku. Tato služba ukazuje, jak lze propojit přehrávání videa s interakcí mezi uživateli, což je klíčový princip celé aplikace PlayStorm.



Obrázek 2 - Watch2Gether Logo [2]

## Technologie

### React & Vite

Moderní knihovna v JavaScriptu, která se používá pro tvorbu uživatelských rozhraní a je postavená na komponentách, díky čemuž je kód přehledný a dobře se udržuje. Pracuje s virtuálním DOM, což umožňuje rychlé a efektivní aktualizace stránky. Vite je nástroj určený pro vývoj a sestavení aplikací, který výrazně zrychluje práci díky okamžitému spuštění projektu a rychlému obnovení změn bez nutnosti zdlouhavého buildování projektu.

### Typescript

Nadstavba jazyka Javascript která nabízí statické typování a další nástroje pro kontrolu kódu a zamezení nekonzistenci typů při práci s daty.

### Tailwind CSS

Moderní CSS Framework který umožnuje úpravu a tvorbu CSS stylů přímo v html kódu bez nutnosti tvorby stylových souborů CSS

### Socket.io

Knihovna využívající webové sokety pro komunikaci v reálném čase mezi klientem a serverem. Umožňuje práci s daty mezi klientem a serverem bez nutnosti obnovování nebo znovu načítání stránky a změna dat se tedy projeví hned jakmile se provede.

### Vidstack

Moderní knihovna pro práci s videem na webu. Slouží jako alternativa k běžným HTML5 přehrávačům. Poskytuje již vytvořené komponenty a API pro snadnou integraci videa/streamů nebo vlastních ovládacích prvků pro aplikaci. Jeho hlavním benefitem je absolutní volnost stylizace přehrávače a nastavení argumentů pro přijímaný obsah.

### Zustand

Knihovna pro správů stavů v aplikaci. Umožňuje ukládat a spravovat globální stavy bez složité konfigurace. Díky jednoduchému API je práce se stavem přehledná a efektivní, což z ní dělá vhodné řešení jak pro menší projekty

### NestJS

Moderní backendový framework postavený na Node.js, který využívá TypeScript a je inspirovaný architekturou Angularu. Je založený na modulárním principu, používá dependency injection a jasně odděluje jednotlivé části aplikace, což výrazně zvyšuje přehlednost a udržovatelnost kódu. NestJS se často používá pro tvorbu REST API nebo real-time serverů, je dobře škálovatelný a vhodný jak pro menší projekty, tak i pro rozsáhlé backendové aplikace.

### Prisma

ORM nástroj pro práci s databázemi v JavaScriptu a TypeScriptu, který výrazně zjednodušuje komunikaci mezi aplikací a databází. Umožňuje definovat databázový model pomocí přehledného schématu, ze kterého automaticky generuje typově bezpečný klient pro práci s daty. Díky tomu je kód čitelnější, méně náchylný k chybám a dobře se používá například v kombinaci s NestJS nebo Express, nejčastěji s databázemi jako PostgreSQL nebo MySQL.

### Postgresql

Open-source relační databázový systém, který se používá pro ukládání a správu dat v moderních aplikacích. Nabízí vysokou spolehlivost, bezpečnost a podporu pokročilých funkcí, jako jsou transakce, indexy nebo práce s relačními daty. Díky dobré škálovatelnosti a kompatibilitě s nástroji jako Prisma nebo backendovými frameworky typu NestJS je PostgreSQL častou volbou pro menší i rozsáhlé projekty.

### Argon 2 & JWT

Technologie používané především v oblasti autentizace a zabezpečení aplikací. Argon2 je moderní a bezpečný algoritmus pro hashování hesel, který chrání uživatelská data tím, že hesla ukládá v nevratné podobě a je odolný vůči útokům hrubou silou. JWT (JSON Web Token) slouží k bezpečnému přenosu informací o uživateli mezi klientem a serverem, nejčastěji ve formě přihlašovacího tokenu. Společně se tyto technologie používají k vytvoření bezpečného přihlašovacího systému v moderních webových aplikacích.

### Go (Golang)

Programovací jazyk vyvinutý společností Google, který je navržený s důrazem na výkon, jednoduchost a efektivní práci s paralelními procesy. Nabízí přehlednou syntaxi, rychlou kompilaci a vestavěnou podporu pro práci s více vlákny pomocí gorutin. Go se často používá pro vývoj backendových služeb, API nebo serverových aplikací, kde je klíčová vysoká rychlost, stabilita a dobrá škálovatelnost.

### FFmpeg

FFmpeg je výkonný open-source nástroj určený pro práci s multimediálním obsahem, především pro zpracování videa a audia. Umožňuje převod formátů, úpravu kvality, extrakci zvuku, práci se streamy nebo základní editaci médií pomocí příkazové řádky. FFmpeg se často používá na serverech a v automatizovaných systémech, kde je potřeba efektivně zpracovávat multimediální soubory.

### Python

Vysokoúrovňový programovací jazyk známý svou jednoduchou a čitelnou syntaxí, díky které je vhodný jak pro začátečníky, tak pro pokročilé vývojáře. Používá se v širokém spektru oblastí, jako je backendový vývoj, automatizace, práce s daty nebo umělá inteligence. Python má rozsáhlý ekosystém knihoven a frameworků, což umožňuje rychlý vývoj aplikací a snadné řešení komplexních úloh.

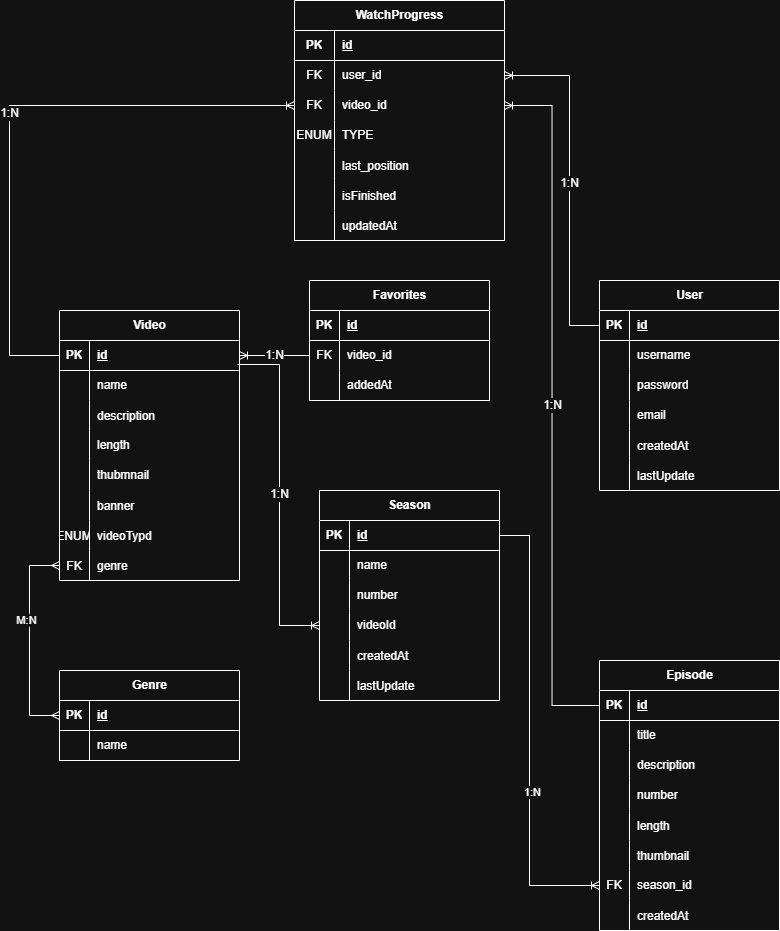
### Docker

Docker je nástroj pro kontejnerizaci aplikací, který umožňuje zabalit aplikaci spolu se všemi jejími závislostmi do jednoho izolovaného prostředí zvaného kontejner. Díky tomu aplikace běží konzistentně na různých systémech bez ohledu na prostředí, ve kterém je spuštěna. Docker se běžně používá pro nasazování aplikací, zjednodušení správy serverů a vývoj moderních backendových i full-stack projektů, kde je důležitá přenositelnost a škálovatelnost.

# Praktická část.

## Návrhy

### 2.1.1 Návrh Databáze

Vytvořil jsem si ERD návrh databáze aby pro mě byla tvorba schéma databáze v Prismě jednodušší a abych mohl lépe pracovat s případnými opravami a úpravami kterých bylo během tvoření projektu nemálo. Do rozložení tohoto návrhu jsem se inspiroval z různých schémat větších streamovacích platforem jako Netflix, HBO apd. Databáze odkazuje spoustu 1:N vztahů což je u těchto streamovacích služeb celkem častý jev. tento návrh mi pomohl při hledání problémů během vývoje backendu a při tvorbě návrhu pro celou restful API. V databázi jsou zvlášť řešené filmy a zvlášť seriály.

Obrázek 3 - ERD Diagram

### 2.1.2 Návrh backend routing a API

### 

## Produktizace

Sem napište popis vybraných částí vašeho kódu a algoritmů.

## Popis pro uživatele

Sem napište uživatelský tutoriál.

Závěr

Sem rozepište zhodnocení cílů ročníkové práce.

POKYNY: V závěru rozeberte, co se vám z výše uvedených cílů povedlo/nepovedlo. NEPOUŽÍVEJTE zde odrážky/body! Většinou se závěr píše tak, že si otevřete Úvod a sepisujete, jak jste naplnili/nenaplnili původní očekávání.

Zároveň sem můžete připsat, co vám projekt dal, kdo vám pomáhal, jestli pro něj máte nějaké plány do budoucna, nebo jestli je/bude někde nasazen, atd.

Rozsahově by měl mít závěr cca **jednu stránku** (jako úvod). NEMÁ obsahovat obrázky ani seznam bodů. MÁ to být jen souvislý text uspořádaný do několika odstavců. Na text odstavců použijte předpřipravený styl s názvem ***Normální text***.

Seznam použitých zdrojů

Sem vygenerujte seznam použité literatury a zdroje obrázků (Reference -> Bibliografie).

POKYNY: Použijte **citační normu ISO 690**. Využijte možnosti citačního manažeru [www.citace.com](http://www.citace.com), případně podrobné vysvětlení s příklady: <https://ezdroje.upol.cz/files/Citovani-CSN_ISO690.pdf> .

Seznam obrázků

Sem vygenerujte seznam obrázků (Reference -> Titulky).

[Obrázek 1 - Netflix Logo [1] 8](#_Toc218245436)

[Obrázek 2 - Watch2Gether Logo [2] 8](#_Toc218245437)

[Obrázek 3 - ERD Diagram 11](#_Toc218245438)

POKYNY: Jednotlivé obrázky označte pomocí automaticky číslovaných **titulků** (Reference – Vložit titulek). Poté vygenerujte seznam obrázků (Reference -> Vložit seznam obrázků -> Popisek titulku: Obrázek). Při generování seznamu ponechte číslování stránek, na kterých se obrázky nachází.

Seznam kódů

Sem vygenerujte seznam kódů (Reference -> Titulky).

POKYNY: Jednotlivé bloky kódu označte pomocí automaticky číslovaných **titulků** (Reference – Vložit titulek). Poté vygenerujte seznam kódů (Reference -> Vložit seznam obrázků -> Popisek titulku: Kód). Při generování seznamu ponechte číslování stránek, na kterých se obrázky nachází.

Po vygenerování upravte styl seznamu tak, aby byl jednotný se stylem textu (doporučuji využívat předpřipravený styl ***Normální***).

Seznam tabulek

Sem vygenerujte seznam tabulek (Reference -> Titulky).

[Tabulka 1: Titulek k vzorové tabulce. Zdroj: autor 8](#_heading=h.erorc61id7fw)

POKYNY: Jednotlivé tabulky označte pomocí automaticky číslovaných **titulků** (Reference – Vložit titulek). Poté vygenerujte seznam tabulek (Reference -> Vložit seznam obrázků -> Popisek titulku: Tabulka). Při generování seznamu ponechte číslování stránek, na kterých se tabulky nachází.

Po vygenerování upravte styl seznamu tak, aby byl jednotný se stylem textu (doporučuji využívat předpřipravený styl ***Normální***).

Pokud Vaše práce neobsahuje žádnou tabulku, tuto kapitolu odstraňte!

Seznam objektů

Sem vygenerujte seznam dalších objektů (Reference -> Titulky).

POKYNY: Pokud Vaše práce obsahuje další grafické objekty (např. grafy, fotografie, audionahrávky, videa, apod.), vytvořte si nový popisek s typem objektu (Reference – Vložit titulek -> Nový popisek), přidejte ke každému objektu titulek (Reference – Vložit titulek) a na tomto místě vložte seznam těchto objektů (stejně jako u obrázků).

Pokud Vaše práce neobsahuje žádné další grafické objekty, tuto kapitolu odstraňte!

Obsah média

Sem vložte popis přiloženého média (jen pokud je součástí práce).

POKYNY: Popište stručně adresářovou strukturu pro všechny důležité soubory (použijte např. víceúrovňový seznam). Pokud médium (CD, DVD, Flashdisk) obsahuje celý projekt s velkým množstvím knihoven, nevypisujte cesty ke všem souborům. Uveďte pouze kde se nachází projekt, kde se nachází build, apod.

Médium fyzicky označte vaším **jménem, třídou, školním rokem**!Médium připojte k dokumentaci tak, aby nevypadávalo, ale aby se zároveň dalo vyndat a použít.

Médium by mělo obsahovat následující části:

* Projekt
* Export databáze (pokud ji obsahuje)
* Spustitelný build (nebo aspoň odkaz, kde se nachází spustitelná verze)
* Dokumentaci v PDF formátu + vhodný další editovatelný formát (docx, odt)
* Prezentaci připravenou k obhajobě

**Závěrečné poznámky (před odevzdáním vymažte):**

* Dokumentace může obsahovat různá poděkování.
* U jednotlivých odstavců textu dokumentace zkontrolujte zarovnání do bloku.
* Před exportem do PDF aktualizujte všechny vygenerované seznamy (obsah, zdroje, seznam obrázků) a zkontrolujte, že je vše v pořádku.
* Před tiskem si dokumentaci vyexportujte do PDF formátu a zkontrolujte odsazení textu a umístění obrázků.
* Dokumentace může být vytisknutá černobíle.
* Dokumentace musí být vytisknutá jednostranně.
* **V pololetí se dokumentace netiskne!**
* Vytištěná dokumentace by měla být svázána kroužkovou vazbou s průhlednou přední stranou a neprůhlednou zadní stranou (barva zadní strany a vazby je na vás).

**V případě dotazů k dokumentaci kontaktujte vedoucího práce nebo vyučujícího předmětu Projekty!**