Paneurópska vysoká škola v Bratislave
Fakulta informatiky

Evidenčné číslo: FI-11145-22777

Štefan Rapčo

Webová aplikácia pre správu úloh

Bakalárska práca

Vedúci práce: RNDr. Ján Lacko, PhD.

Máj 2025

Paneurópska vysoká škola v Bratislave

Fakulta informatiky

Evidenčné číslo: FI-11145-22777

Štefan Rapčo

Webová aplikácia pre správu úloh

Bakalárska práca

Študijný program: Aplikovaná Informatika

Študijný odbor: Informatika

Školiace pracovisko: Ústav aplikovanej informatiky

Vedúci práce: RNDr. Ján Lacko, PhD.

Máj 2025

PANEURÓPSKA VYSOKÁ ŠKOLA

Fakulta informatiky

ZADANIE BAKALÁRSKEJ PRÁCE

Meno a priezvisko študenta:	**Štefan Rapčo**
Evidenčné číslo bakalárskej práce:	**FI-11145-22777**
Študijný odbor:	**informatika**
Študijný program:	**Aplikovaná informatika**
Forma a metóda štúdia:	**denná prezenčná**
Vedúci bakalárskej práce:	**RNDr. Ján Lacko, PhD.**
Ústav/katedra:	**Ústav aplikovanej informatiky**
Dátum zadania bakalárskej práce:	**05. 09. 2024**

Názov: **Webová aplikácie pre správu úloh**

Anotácia: Cieľom práce je návrh, implementácia a testovanie webovej aplikácie pre správu úloh (Task manager). V teroretickej časti práce študent popíše východiská pre použité technológie a porovná konkurenčné riešenia. V praktickej časti navrhne, impelementuje a otestuje webovú aplikáciu s možnosťou správy používateľov, kategorizácie taskov, ich životného cyklu s možnosťou spolupráce viacerých používateľov.

RNDr. Ján Lacko, PhD.
vedúci práce

Ing. Juraj Štefanovič, PhD.
vedúci ústavu

Čestné prehlásenie

Čestne vyhlasujem, že som túto prácu vypracoval samostatne, na základe konzultácií a s použitím uvedenej literatúry.

V Bratislave, 21.5.2025 .....

Štefan Rapčo

Poďakovanie

Rád by som sa poďakoval pánovi RNDr. Ján Lacko, PhD. za všetky spoločné konzultácie a rady, bez ktorých by som túto tému sám nevypracoval.

Anotácia

Paneurópska vysoká škola v Bratislave
Fakulta informatiky
Študijný program: Aplikovaná Informatika

Autor: Štefan Rapčo
Bakalárska práca: Webová aplikácia pre správu úloh
Vedúci projektu: RNDr. Ján Lacko, PhD.
Máj 2025

Annotation

Pan-European University in Bratislava
Faculty of informatics
Degree Course: Applied Informatics

Author: Štefan Rapčo
Bachelor's thesis: Web application Task Manager
Supervisor: RNDr. Ján Lacko, PhD.
May 2025

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam použitých skratiek a anglicizmov

FE	Frontend
BE	Backend
DB	Databáza
OWASP	Open Worldwide Application Security Project
REST	Representational State Transfer
API	Application Programming Interface
UI	User Interface
DOM	Document Object Model
ORM	Object-Relational Mapper
SQL	(Structured Query Language)
ŠPZ	Štátna Poznávacia Značka
Stack	Súbor technológií, ktoré spolupracujú na vytvorení aplikácie. Každá aplikácia pozostáva z viacerých vrstiev. Stack definuje, aké konkrétne technológie sú použité v každej z týchto vrstiev
Runtime	Obdobie, počas ktorého je program spustený a vykonáva svoje inštrukcie

GraphQL Cacheovanie	Mechanizmus, ktorý pomáha optimalizovať výkon aplikácií. Apollo Client používa cacheovanie na ukladanie dát, ktoré boli získané prostredníctvom GraphQL požiadaviek, a umožňuje aplikácii opätovne použiť tieto dáta bez toho, aby musela posielať novú požiadavku na server
Asynchronicita	Pojem, ktorý označuje spôsob vykonávania operácií, pri ktorých nečakáme na dokončenie jednej operácie predtým, ako začneme vykonávať ďalšiu
Event-driven	Vzor alebo prístup, ktorý sa zameriava na reakciu na udalosť alebo zmenu stavu v systéme
Chat	Bežné označenie pre platformy, kde si ľudia vymieňajú správy v reálnom čase
Injection	Technika, kde útočník vkladá škodlivý kód do aplikácie alebo systému, aby spôsobil nežiaduce správanie

1. Úvod

Pri návrhu a vývoji webovej aplikácie na správu úloh je dôležité najskôr pochopiť samotný problém, ktorý má riešiť, a následne zvoliť technológie, ktoré umožnia jej efektívnu realizáciu. Táto práca sa preto v úvodných kapitolách venuje analýze problému, definovaniu požiadaviek na aplikáciu a výberu technologického stacku, ktorý zabezpečí jej spoľahlivú a výkonnú prevádzku.

V rámci analýzy boli preskúmané existujúce riešenia, ako Notion, Asana a Microsoft To Do, pričom sme sa zamerali na ich prístup k správe úloh a technológie, na ktorých sú postavené. Toto porovnanie pomohlo lepšie pochopiť výhody a nevýhody konkurenčných aplikácií a inšpirovať sa overenými riešeniami, ktoré zlepšujú používateľskú skúsenosť.

Na základe tejto analýzy bola vybraná kombinácia moderných a široko používaných technológií, ktoré sú spoľahlivé, výkonné a osvedčené v priemysle. React, TypeScript, Apollo GraphQL, Node.js, MySQL a Prisma poskytujú stabilný základ pre aplikáciu, ktorá ponúka intuitívne používateľské rozhranie, efektívnu manipuláciu s dátami a výkonnú BE architektúru.

Táto práca ponúka komplexný pohľad na problematiku správy úloh, hodnotenie existujúcich riešení a vysvetlenie výberu technológií, ktoré umožnia vytvoriť stabilnú, bezpečnú a výkonnú aplikáciu.

2. Analýza problému

Pred samotnou implementáciou je dôležité analyzovať existujúce riešenia na trhu, identifikovať ich silné a slabé stránky a porovnať ich s plánovanými funkciami navrhovanej aplikácie.

2.1 Prehľad existujúcich aplikácií na správu úloh

V dnešnej dobe existuje množstvo digitálnych aplikácií na riadenie úloh, ktoré umožňujú používateľom efektívne organizovať pracovné aj osobné aktivity. Medzi najznámejšie aplikácie v tejto oblasti patria Asana, Notion a Microsoft To Do.

Každá z nich ponúka špecifické funkcie a prístupy k správe úloh, vďaka čomu sú vhodné pre rôzne typy používateľov – od jednotlivcov až po veľké tímy.

Pozrime sa na ich špeficikácie:

1. **Asana**: Asana patrí medzi najvýkonnejšie aplikácie na správu projektov, zamerané predovšetkým na tímy a organizácie. Aplikácia umožňuje vytvárať a spravovať úlohy v rámci projektov, nastavovať termíny, delegovať úlohy členom tímu a sledovať pokrok prostredníctvom rôznych vizuálnych reprezentácií, ako sú zoznamy, tabuľky a časové osi [1].

   Výhodou aplikácie Asana je aj schopnosť používateľom umožniť pridávať

podúlohy, komentovať úlohy, nastavovať závislosti medzi nimi a spolupracovať s množstvom externých nástrojov, ako sú Slack, Google Drive či Microsoft Teams [2].

Pre veľké organizácie je však nevýhodou pomerne vysoká cena prémiových funkcií a zložitejšie učenie sa aplikácie pre úplných začiatočníkov [1].

2. **Notion**: Notion predstavuje komplexnú aplikáciu, ktorá kombinuje správu úloh, poznámky, databázy a spoluprácu v jednom rozhraní. Na rozdiel od ostatných aplikácií, ktoré sa sústreďujú výhradne na úlohy a projekty, Notion umožňuje vytvárať prispôsobiteľné stránky, ktoré môžu obsahovať zoznamy úloh, tabuľky, kalendáre a rôzne multimediálne prvky [3] [4].

   Silnou stránkou aplikácie Notion je flexibilita – používateľ si môže prispôsobiť štruktúru aplikácie presne podľa svojich potrieb [3]. To však znamená, že to môže byť zároveň aj nevýhoda, pretože na plné využitie všetkých možností si nový používateľ musí osvojiť relatívne komplexný spôsob práce.

3. **Microsoft To Do**: Microsoft To Do je aplikácia na správu úloh, ktorá sa prepája s ekosystémom Microsoft 365. Používateľom umožňuje vytvárať zoznamy úloh, nastavovať termíny a pripomienky, pridávať poznámky a kategorizovať úlohy podľa farieb a značiek [5] [6].

   Hlavnou výhodou Microsoft To Do je hlboká spolupráca so službou Microsoft Outlook a ďalšími Microsoft službami, čo ho robí ideálnou aplikáciou pre firemných používateľov, ktorí už pracujú v prostredí Microsoft služieb [5].

   Z tohto nám vyplýva, že v porovnaní s aplikáciou Asana, Microsoft To Do neposkytuje až také pokročilé projektové funkcie.

2.2 Kľúčové funkcionality navrhovanej aplikácie

Pozrime sa na kľúčové funkcionality, ktoré navrhovaná aplikácia musí spĺňať:

1. Overenie používateľa:
    - Bezpečné a overené prihlasovanie a registrácia
    - Obnovenie hesla a overenie e-mailu
2. Správa úloh:
    - Vytvárať, upravovať a odstraňovať úlohy
    - Stanoviť priority a termínu pre úlohu, usporiadať úlohy do prispôsobiteľných kategórií
    - Filtrovanie a vyhľadávanie úloh na základe rôznych kritérií
3. Spolupráca:
    - Zdieľanie úloh s ostatnými používateľmi
    - Prideľovanie úloh iným registrovaným používateľom v rovnakom projekte, možnosť pridať komentár ku úlohe
4. Oznámenia:
    - E-mailové upozornenia a nastavenia upozornení
5. Analytická časť:
    - Prehľad úloh
    - Vizuálne správy o dokončení úloh
    - Zobrazenie grafov a tabuliek, zhrnutie času koľko aký používateľ strávil čas na daných úlohách (možné filtrovanie)

2.3 Porovnanie funkcionality s existujúcimi riešeniami

V súčasnosti existuje veľa aplikácií na správu úloh, no každá z nich pristupuje k organizácii práce trochu inak. Niektoré sa viac sústredia na individuálnu produktivitu, iné na tímové riadenie alebo integrácie s inými službami.

Aby sme lepšie pochopili jej výhody a možné oblasti na zlepšenie, porovnáme ju s najznámejšími riešeniami spomenutými vyššie.

2.3.1 Overenie používateľa – ako sa prihlásiť bezpečne

Navrhovaná aplikácia kladie dôraz na bezpečnosť používateľov. Poskytuje registráciu, prihlásenie, obnovu hesla a e-mailové overenie, čím zabezpečuje ochranu dát a zabraňuje neoprávnenému prístupu. Tento prístup pomáha udržať používateľské účty v bezpečí a zároveň zvyšuje dôveru v aplikáciu.

V tomto smere sa podobá na nájstroj Asana a Microsoft To Do, ktoré tiež dbajú na bezpečnosť. Tieto platformy však ponúkajú aj Single Sign-On (SSO), čo je veľká výhoda hlavne pre firemných používateľov [7] [8].

- Výhoda oproti konkurencii: Možnosť overenia cez e-mail pri registrácii, čo nie všetky aplikácie ponúkajú
- Nevýhoda oproti konkurencii: Absencia SSO, ktoré sú štandardom v Asane a Microsoft To Do

2.3.2 Správa úloh – čo s nimi vieme robiť

Každá aplikácia na správu úloh má základné funkcie, ako je vytváranie, úprava a mazanie úloh. Rozdiel je však v tom, ako sú úlohy organizované a aké možnosti má používateľ na ich spravovanie.

Asana má pokročilejšie možnosti nastavovania priorít a termínov, pričom Asana umožňuje aj prepojenie medzi úlohami a vytváranie závislostí medzi nimi [2].

- Výhoda oproti konkurencii: Flexibilné kategórie a detailné filtrovanie úloh, čo nie je v každej aplikácii samozrejmosťou
- Nevýhoda oproti konkurencii: Chýba automatizácia pracovných úloh, ktorú má Asana

2.3.3 Spolupráca – ako pracovať v tíme

Jednou z dôležitých vlastností aplikácií na správu úloh je podpora tímovej práce. Navrhovaná aplikácia umožňuje zdieľanie úloh, prideľovanie členom tímu a komentovanie, čím podporuje základnú spoluprácu.

Na druhej strane, niektoré konkurenčné aplikácie to majú oveľa lepšie. Asana a Notion ponúkajú diskusné "vlákna" a správy priamo v rámci úloh, čím umožňujú podrobnejšiu komunikáciu bez nutnosti používať externé nástroje [1]. Microsoft To Do je zase pevne prepojený s Microsoft Teams [6], z čoho nám vyplýva, že aplikácia je vhodná aj pre firemné prostredie.

- Výhoda oproti konkurencii: Základná tímová práca a zdieľanie úloh
- Nevýhoda oproti konkurencii: Chýbajú pokročilé diskusné funkcie a integrácia s nástrojmi ako Slack či Microsoft Teams

2.3.4 Oznámenia – aby sme na nič nezabudli

Každá aplikácia na správu úloh ponúka nejaký spôsob upozornení. Navrhovaná aplikácia využíva e-mailové notifikácie, ktoré si používatelia môžu prispôsobiť podľa svojich potrieb. To predstavuje významnú výhodu oproti niektorým riešeniam, kde sú notifikácie odosielané automaticky, bez možnosti ich nastavenia či úpravy.

Na druhej strane, konkurencia ako Asana a Microsoft To Do ponúka aj mobilné notifikácie, čím používateľom okamžite pripomenie dôležité úlohy [1].

- Výhoda oproti konkurencii: Možnosť si prispôsobiť, aké e-mailové notifikácie budú používatelia prijímať
- Nevýhoda oproti konkurencii: Chýbajú mobilné notifikácie

2.3.5 Analytika – ako sledovať postup

Jedným z hlavných rozdielov medzi navrhovanou aplikáciou a konkurenciou je analytika. Aplikácia ponúka podrobné prehľady o dokončených úlohách a vizualizácie produktivity.

V tomto smere prekonáva niektoré základné aplikácie, ako je Microsoft To Do, ktoré majú len jednoduché štatistiky. Na druhej strane, Microsoft To Do umožňuje vďaka svojmu ekosystému s Microsoft aplikáciami export dát do Power BI pre podrobnejšiu analýzu, čo je výhoda pre veľké firmy [6].

- Výhoda oproti konkurencii: Detailné sledovanie času stráveného na úlohách na úrovni jednotlivých používateľov
- Nevýhoda oproti konkurencii: Chýba prepojenie s analytickými nástrojmi ako Power BI

Funkcia	Navrhovaná aplikácia	Asana	Notion	MS To Do
Overenie e-mailom	✓	✓	✓	✓
SSO	×	✓	✓	✓
2FA	✓	✓	×	✓
Pokročilé filtrovanie	✓	✓	✓	×
Vhodné pre začiatočníka	✓	×	×	✓
Tímová spolupráca	✓	✓	✓	✓
Mobilné notifikácie	×	✓	✓	✓
Minimalistický dizajn	✓	×	×	✓
Profesionálne aplikovanie analýzy	×	✓	×	×
Export dát do Power BI	×	×	×	✓

Table 2.1: Porovnanie kľúčových funkcií jednotlivých task managerov

Na základe vykonanej analýzy možno povedať, že navrhovaná aplikácia kombinuje rôzne prvky existujúcich aplikácií na správu úloh. Využíva ich silné stránky, no zároveň čelí určitým výzvam, ktoré bude potrebné riešiť. Nejde o kópiu žiadneho konkrétneho riešenia, ale o hybridný systém, ktorý prepája výhody viacerých populárnych aplikácií do jedného celku.

Z Notionu si aplikácia berie flexibilitu v organizovaní úloh, čo umožňuje používateľom triediť úlohy do prispôsobiteľných kategórií a filtrovať ich podľa rôznych kritérií. Analytika a vizuálne prehľady o produktivite zase pripomínajú Asanu, pričom v niektorých aspektoch umožňujú detailnejšie sledovanie času stráveného na jednotlivých úlohách.

Aplikácia zároveň kladie dôraz na jednoduchosť a efektivitu v správe úloh, pričom si zachováva minimalistický dizajn, ktorý podporuje osobnú produktivitu aj tímovú spoluprácu. Pokiaľ ide o bezpečnosť a správu prístupu, približuje sa k Microsoft To Do, hoci neponúka integráciu s ekosystémom Microsoft 365.

Celkovo možno aplikáciu vnímať ako kombináciu silných stránok viacerých existujúcich riešení, no zároveň s priestorom na vylepšenia, ktoré by ju mohli ešte viac priblížiť potrebám používateľov.

3. Použité technológie a porovnanie s inými aplikáciami

V tejto kapitole sa zameriame na technologický stack použitý pri vývoji aplikácie. Pre lepšiu prehľadnosť sme ho rozdelili do troch hlavných častí:

- **Frontend**: Zaoberá sa vizuálnou stránkou aplikácie a interakciou s používateľom.
- **Backend**: Spracúva logiku aplikácie, obsluhuje požiadavky a komunikuje s databázou.
- **Databáza a ORM**: Umožňuje ukladanie a manipuláciu s údajmi aplikácie.

3.1 Frontend

Frontend je viditeľná časť aplikácie, cez ktorú používatelia interagujú. Zodpovedá za vzhľad, ovládanie a používateľskú skúsenosť, pričom komunikuje s BE cez API na získavanie a odosielanie dát.

V tejto kapitole sa pozrieme na kľúčové technológie, ktoré umožňujú vytvárať dynamické, responzívne a výkonné webové aplikácie, prispôsobené rôznym zariadeniam a používateľským potrebám.

3.1.1 React

React je populárna a výkonná JavaScriptová knižnica, ktorú vyvinula spoločnosť Meta (predtým Facebook) na tvorbu moderných a interaktívnych webových aplikácií. Využíva komponentový prístup, čo znamená, že jednotlivé prvky UI môžu byť opakovane použité v rôznych častiach aplikácie. Tento prístup zjednodušuje správu kódu, zvyšuje jeho prehľadnosť a urýchľuje vývoj, čo umožňuje efektívnejšiu prácu na rozsiahlých projektoch [9].

Jednou z hlavných výhod Reactu je virtuálny DOM, ktorý inteligentne aktualizuje iba tie časti stránky, ktoré sa zmenili, čím zlepšuje výkon a plynulosť aplikácie [9]. Vďaka týmto vlastnostiam sa React stal jednou z najpopulárnejších technológií na vývoj interaktívnych a dynamických webových aplikácií, kde je dôležitá rýchlosť a plynulá používateľská skúsenosť.

3.1.2 Typescript

TypeScript je nadstavba JavaScriptu, ktorá poskytuje a pridáva statické typovanie, čím pomáha vývojárom písať spoľahlivejší a prehľadnejší kód. [10]. Vďaka tomu dokáže odhaliť chyby už počas vývoja, namiesto toho, aby sa objavili až pri spustení aplikácie. To výrazne znižuje množstvo runtime chýb a zvyšuje stabilitu kódu.

Okrem bezpečnosti TypeScript podporuje a uľahčuje spoluprácu v tímoch. Typové anotácie robia kód čitateľnejším a jednoduchším na údržbu, pretože jasne definujú, aké dáta a štrukrúry dát sa v aplikácii používajú. To vedie k efektívnejšiemu a bezpečnejšiemu vývoju, najmä pri veľkých a časom sa vyvíjajúcich projektoch [10].

3.1.3 MUI (Material-UI)

MUI je populárna knižnica UI komponentov pre React, ktorá je založená na Material Design princípoch od Google. Tieto princípy sa zameriavajú na to, aby sa aplikácie a webové stránky stali intuitívnejšími, príjemnejšími a konzistentnejšími. Poskytuje širokú škálu predpripravených komponentov, ako sú tlačidlá, formuláre, tabuľky či navigačné panely, vďaka čomu vývojári nemusia tvoriť rozhranie od nuly. [11]. To zrýchľuje vývoj a zaisťuje jednotný dizajn aplikácie. Pre nás to znamená, že sa nemusíme zamýšľať nad dizajnom až do úplného extrému.

Jednou z hlavných výhod MUI je jeho vysoká prispôsobiteľnosť. To znamená, že umožňuje jednoduchú úpravu vzhľadu a štýlu komponentov tak, aby zodpovedali vizuálnej identite aplikácie [11]. Vďaka tomu je možné rýchlo vytvárať moderné a esteticky zladené používateľské rozhrania bez obetovania flexibility či originality dizajnu.

3.1.4 Apollo GraphQL Client

Apollo Client je výkonná knižnica pre komunikáciu s GraphQL API, ktorá zjednodušuje načítavanie, cachovanie a spracovanie dát na strane klienta. Na rozdiel od tradičných REST API, kde klient často prijíma aj nadbytočné informácie, GraphQL umožňuje získať presne tie dáta, ktoré sú potrebné [12]. Toto minimalizuje dátový prenos a zrýchľuje fungovanie aplikácie.

Predstavme si model autíčka, ktorý obsahuje presne definované atribúty, ako sú model, farba, značka, ŠPZ, či počet miest na sedenie. V závislosti od konkrétneho použitia však používateľ nemusí vždy potrebovať všetky tieto údaje. Pri tradičnom REST API by BE vrátil celý objekt autíčka, aj keď FE potrebuje len jednu konkrétnu informáciu, napríklad farbu. Tento spôsob môže viesť k zbytočne

veľkému prenosu dát, čo môže negatívne ovplyvniť výkon aplikácie a zbytočne zaťažiť sieť. Naopak, Apollo Client umožňuje klientovi presne definovať, ktoré údaje chce načítať. Ak FE vyžaduje iba farbu autíčka, Apollo Client zabezpečí, že BE odošle iba túto jednu hodnotu. Tento prístup minimalizuje prenášané dáta a zvyšuje efektivitu komunikácie medzi FE a BE a zlepšuje celkový výkon webovej aplikácie.

Okrem efektívneho získavania dát Apollo Client obsahuje aj zabudované riešenie na správu stavu aplikácie, takže vývojári nemusia používať externé knižnice ako Redux či MobX. Vďaka automatickému cachovaniu odpovedí znižuje počet sieťových požiadaviek, čo optimalizuje výkon aplikácie a zlepšuje používateľskú skúsenosť [12]. Tieto vlastnosti robia Apollo Client ideálnym nástrojom pre vývoj moderných a efektívnych webových aplikácií.

3.2 Backend

Backend si môžeme predstaviť mozog aplikácie, ktorý spracováva požiadavky z FE, riadi obchodnú logiku a komunikuje s DB. Zabezpečuje, aby sa dáta správne ukládali, načítavali a spracovávali, pričom dbá na bezpečnosť a spoľahlivosť celého systému.

Táto časť aplikácie je kľúčová pre jej výkon a škálovateľnosť, pretože musí efektívne zvládať množstvo súbežných požiadaviek. V tejto kapitole sa pozrieme na technológie, ktoré umožňujú BE fungovať stabilne, bezpečne a efektívne aj pri väčšej záťaži.

3.2.1 Node.js

Node.js je výkonné JavaScriptové prostredie, ktoré umožňuje spúšťať kód na serveri, namiesto toho, aby bežal iba v prehliadači. Je postavené na V8 engine od Google Chrome, čo mu poskytuje vysokú rýchlosť a efektivitu pri vykonávaní kódu [13].

Jeho asynchrónna a event-driven architektúra [13] ho robí ideálnym riešením pre aplikácie, ktoré potrebujú spracovávať veľké množstvo súbežných požiadaviek, ako sú chaty, aplikácie v reálnom čase alebo API servery.

3.2.2 Apollo GraphQL Server

Apollo Server je výkonný nástroj na tvorbu GraphQL API v Node.js, ktorý umožňuje efektívne spracovanie dotazov a mutácií. Na rozdiel od tradičných REST API, kde klient často dostáva aj nadbytočné dáta, GraphQL umožňuje precízne špecifikovať, aké informácie sa majú načítať [14]. Túto tému sme už opísali v podkapitole Apollo GraphQL Client.

Apollo Server ponúka schémovú validáciu, ktorá zabezpečuje, že dáta majú vždy správnu štruktúru, autentifikáciu, ktorá chráni prístup k citlivým informáciám, a cachovanie, ktoré znižuje záťaž na databázu a zrýchľuje odpovede na často opakované dotazy [14]. Vďaka týmto vlastnostiam je Apollo Server ideálnou voľbou na budovanie moderných, škálovateľných a efektívnych BE riešení.

3.3 Databáza a ORM

Databáza je centrálne úložisko všetkých údajov aplikácie. Bez spoľahlivej databázy by aplikácia nemohla uchovávať dôležité informácie, ako sú používateľské dáta, nastavenia alebo záznamy o aktivitách.

ORM slúži ako most medzi DB a BE, čím umožňuje vývojárom pracovať s DB pomocou objektovo orientovaných modelov namiesto písania zložitých SQL dotazov. Týmto sa zjednodušuje správa dát, minimalizuje riziko chýb a zvyšuje bezpečnosť webovej aplikácie.

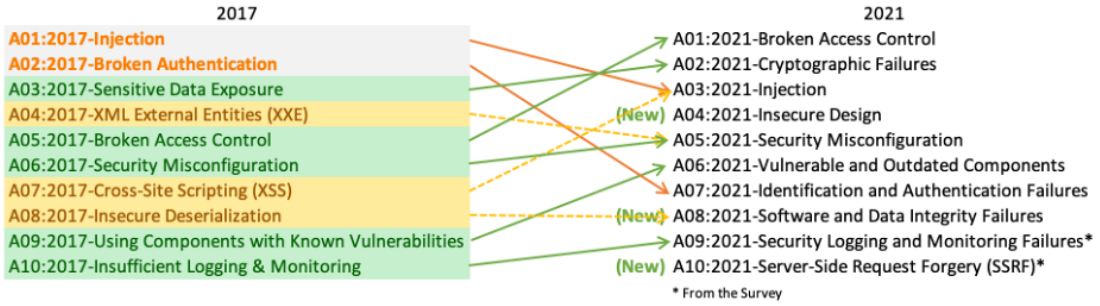
V tejto kapitole sa pozrieme na použité technológie, ktoré umožňujú spoľahlivú manipuláciu s údajmi, a vysvetlíme, ako prispievajú k plynulej a efektívnej prevádzke aplikácie.

3.3.1 Prisma ORM

Prisma je moderný ORM pre Node.js a TypeScript, ktorý vývojárom uľahčuje prácu s databázou. Namiesto písania zložitých SQL dotazov umožňuje používať jednoduché a intuitívne modely, vďaka čomu je práca s dátami prehľadnejšia a efektívnejšia. Prisma tiež podporuje automatické migrácie databázovej schémy, takže úpravy štruktúry databázy sú rýchle a bez potreby manuálneho písania SQL príkazov [15].

Jednou z najväčších výhod Prismi je automatická ochrana pred SQL injection útokmi [16]. Keďže Prisma generuje bezpečné SQL dotazy a dôsledne pracuje so zabezpečenými vstupmi, minimalizuje riziko neoprávneného prístupu k databáze. Táto vlastnosť z nej robí spoľahlivý nástroj na budovanie bezpečných aplikácií. [15]. Je samozrejme nevyhnutné, aby bola aplikácia dôkladne chránená proti SQL injection útokom, keďže patria medzi najzávažnejšie bezpečnostné hrozby pre databázové systémy [17][18], a podľa OWASP boli v roku 2021 tretím najčastejším útokom na databázové systémy [19].

Okrem bezpečnosti Prisma ponúka aj jednoduchšiu správu databázových modelov, automatické generovanie typov a lepší prehľad o dátach [15]. Vďaka týmto vlastnostiam je skvelou voľbou pre moderné aplikácie, ktoré potrebujú stabilné,



Obrázok 3.1: 10 najčastejších webových útokov a rizík [2]OWASP, OWASP Top 10 Web Application Security Risks, 2021, USA, https://owasp.org/www-project-top-ten/, Naposledy navštívené: 5. Február 2025

výkonné a bezpečné databázové riešenie.

3.3.2 MySQL

MySQL je výkonná a spoľahlivá relačná databáza, ktorá patrí medzi najpoužívanejšie databázové riešenia v priemysle. Využíva SQL na efektívne ukladanie, spracovanie a načítavanie dát, pričom je optimalizovaná na vysoký výkon pri čítaní a aj zápise [20].

Jednou z hlavných výhod MySQL je podpora transakcií, ktoré zabezpečujú konzistenciu údajov (ak nejaký príkaz v transakcii zlyhá, DB je vrátená do pôvodného stavu pred začatím transakcie), indexovania, ktoré zrýchľuje vyhľadávanie, a replikácie, ktorá umožňuje rozloženie záťaže a škálovanie databázy pre veľké aplikácie [20]. Vďaka týmto vlastnostiam je MySQL široko využívané.

3.4 Správa verzií kódu

3.4.1 Git

Git je výkonný verzovací systém, ktorý umožňuje efektívne sledovanie zmien v kóde, čím zabezpečuje organizovanú a transparentnú spoluprácu medzi vývojármi. Vďaka distribuovanej architektúre si každý vývojár udržiava vlastnú kópiu projektu, čo umožňuje pracovať nezávisle a zároveň jednoducho synchronizovať zmeny s ostatnými [21].

Nástroje systému Git umožňujú efektívne spravovať repozitáre a sledovať históriu zmien [21], čo vedie k spoľahlivejšiemu a efektívnejšiemu dodávaniu softvéru.

3.5 Porovnanie technológií

Nasledujúca tabuľka porovnáva použité technológie navrhovanej aplikácie s populárnymi riešeniami, ako sú Notion, Asana a Microsoft To Do.

Technológia	**Navrhovaná aplikácia**	**Notion**	**Asana**	**MS To Do**
React.js	✓	✓	✓	?
TypeScript	✓	✓	✓	?
MUI	✓	×	×	?
Apollo GraphQL	✓	×	✓	?
Node.js	✓	✓	?	?
Databáza	MySQL	PostgreSQL	LunaDB	?
ORM	Prisma ORM	?	?	?
Git	✓	✓	?	?

Table 3.1: Porovnanie použitých technológií mojej aplikácie s Notion, Asana a Microsoft To Do. [22] [23] [24]

Microsoft To Do nanešťastie nepublikuje detaily o svojom stacku, no je pravdepodobné, že využíva vlastné technológie, ako .NET, Azure a MS SQL Server.

Výber technologického stacku pre túto aplikáciu nebol náhodný. Okrem osobnej skúsenosti s týmito technológiami zohrala dôležitú úlohu aj ich osvedčená spoľahlivosť a široké využitie v priemysle. Technológie ako React, TypeScript, Apollo GraphQL, Node.js, MySQL a Prisma patria medzi populárne a overené riešenia, ktoré používajú popredné svetové spoločnosti a aplikácie, vrátane aj Notion a Asana, ktoré sme v tejto kapitole analyzovali.

Každá z týchto technológií ponúka konkrétne výhody, vďaka ktorým je aplikácia výkonná, bezpečná a dobre škálovateľná. React a MUI umožňujú rýchlu tvorbu moderného a flexibilného používateľského rozhrania, TypeScript zvyšuje spoľahlivosť kódu vďaka statickému typovaniu, Apollo GraphQL optimalizuje prenos dát a znižuje zbytočné sieťové požiadavky, zatiaľ čo Node.js poskytuje efektívny BE ekosystém. MySQL a Prisma zabezpečujú stabilné, bezpečné a výkonné databázové riešenie.

Tieto technológie sú dlhodobo overené a široko podporované komunitou aj priemyslom, čo zaručuje stabilitu a dlhodobú udržateľnosť aplikácie. Ich výber teda nebol len otázkou preferencií, ale premysleným rozhodnutím, ktoré vychádza z ich reálneho využitia v profesionálnych projektoch po celom svete.

4. Prehľad aplikácie

TODO:

Záver

TODO:

Zoznam použitej literatúry

[1] Alexander Newnham. "Asana vs Trello vs Todoist – Which is the Best? 2021 Review". In: *Pressidium* (2023). Naposledy navštívené: 29. November 2024. URL: https://pressidium.com/blog/asana-vs-trello-vs-todoist-which-is-the-best-2021-review/.

[2] Ryan Kane. "Asana vs. Todoist: Which task management app should you choose? [2025]". In: *_zapier* (2024). Naposledy navštívené: 29. November 2024. URL: https://zapier.com/blog/asana-vs-todoist/.

[3] Notion Labs Inc. "What is Notion?" In: *Notion* (2025). Naposledy navštívené: 17. Január 2025. URL: https://www.notion.com/help/guides/what-is-notion.

[4] Emma Roth. "Notion is making a super customizable email app". In: *The Verge* (2024). Naposledy navštívené: 17. Január 2025. URL: https://www.theverge.com/2024/10/24/24278848/notion-super-customizable-email-app.

[5] Microsoft Corporation. "Aplikácia Microsoft To Do". In: *Microsoft Corporation* (2025). Naposledy navštívené: 2. Február 2025. URL: https://www.microsoft.com/sk-sk/microsoft-365/microsoft-to-do-list-app.

[6] Jose Rodríguez. "A Comparative Review of Top To-Do List Tools for Boosting Productivity 2024". In: *The Productivity Blog* (2024). Naposledy navštívené:

29. November 2024. URL: https://blog.productivity.directory/a-comparative-review-of-top-to-do-list-tools-for-boosting-productivity-2024-d5368b50ff44.

[7] Asana Inc. "OneLogin + Asana". In: *Asana* (2025). Naposledy navštívené: 2. Február 2025. URL: https://asana.com/apps/onelogin.

[8] Microsoft Corporation. "Microsoft Entra single sign-on (SSO)". In: *Microsoft Corporation* (2025). Naposledy navštívené: 2. Február 2025. URL: https://www.microsoft.com/en-us/security/business/identity-access/microsoft-entra-single-sign-on.

[9] Meta. "React - official documentation". In: *React* (2025). Naposledy navštívené: 4. Február 2025. URL: https://react.dev/.

[10] Microsoft Corporation. "TypeScript - official documentation". In: *Microsoft Corporation* (2025). Naposledy navštívené: 4. Február 2025. URL: https://www.typescriptlang.org/docs/.

[11] MUI Inc. "MUI - official documentation". In: *MUI* (2025). Naposledy navštívené: 4. Február 2025. URL: https://mui.com/material-ui/getting-started/.

[12] Apollo GraphQL. "Apollo GraphQL Client - official documentation". In: *Apollo GraphQL* (2025). Naposledy navštívené: 4. Február 2025. URL: https://www.apollographql.com/docs/react.

[13] Node.js Foundation. "Node.js - official documentation". In: *Node.js Foundation* (2025). Naposledy navštívené: 5. Február 2025. URL: https://nodejs.org/docs/latest/api/documentation.html.

[14] Apollo GraphQL. "Apollo GraphQL Server - official documentation". In: *Apollo GraphQL* (2025). Naposledy navštívené: 5. Február 2025. URL: https://www.apollographql.com/docs/apollo-server.

[15] Prisma. "Prisma ORM - official documentation". In: *Prisma* (2025). Naposledy navštívené: 5. Február 2025. URL: https://www.prisma.io/docs.

[16] Prisma. "Prisma ORM - official documentation - SQL injection prevention". In: *Prisma* (2025). Naposledy navštívené: 7. Február 2025. URL: https://www.prisma.io/docs/orm/prisma-client/using-raw-sql/raw-queries#sql-injection-prevention.

[17] Peeyush Sharma. "A Deep Dive into OWASP Top 3 Security Risks". In: (2023). Naposledy navštívené: 3. Február 2025. URL: https://opencoursehub.cs.sfu.ca/bfraser/grav-cms/cmpt415/report/sample/OWASP_Top3SecurityRisks-HaitiHHA.pdf.

[18] William G Halfond, Jeremy Viegas, Alessandro Orso, et al. "A classification of SQL-injection attacks and countermeasures". In: *Proceedings of the IEEE international symposium on secure software engineering*. Vol. 1. Naposledy navštívené: 3. Február 2025. IEEE. 2006. URL: https://sites.cc.gatech.edu/fac/Alex.Orso/papers/halfond.viegas.orso.ISSSE06.pdf.

[19] OWASP. "OWASP TOP TEN". In: *OWASP* (2021). Naposledy navštívené: 5. Február 2025. URL: https://owasp.org/www-project-top-ten/.

[20] Oracle Corporation. "MySQL - official documentation". In: *Oracle Corporation* (2025). Naposledy navštívené: 5. Február 2025. URL: https://dev.mysql.com/doc/.

[21] Linus Torvalds. "Git - official documentation". In: *Linus Torvalds* (2025). Naposledy navštívené: 5. Február 2025. URL: https://git-scm.com/doc.

[22] Asana Inc. Engineering Team. "Asana's Tech Stack: How we build a collaborative app for teams of all sizes". In: *Asana* (2022). Naposledy navštívené: 6. Február 2025. URL: https://asana.com/inside-asana/asana-tech-stack.

[23] Michael. "Breaking Down Notion's Tech Stack". In: *slashdev* (2025). Naposledy navštívené: 6. Február 2025. URL: https://slashdev.io/-breaking-down-notions-tech-stack.

[24] Himalayas. “Notion Tech Stack”. In: *Himalayas* (2025). Naposledy navštívené: 6. Február 2025. URL: https://himalayas.app/companies/notion/tech-stack.

5. Technická dokumentácia

TODO:

6. Harmonogram práce

TODO:

7. Obsah digitálneho média

Evidenčné číslo práce v informačnom systéme: FI-11145-22777

Obsah digitálnej časti práce (archív ZIP):

Názov odovzdaného archívu: BP_StefanRapco.zip.