# Technologie pro publikování na webu 2

# Semestrální projekt – webová aplikace

**Identifikační údaje**

|  |  |
| --- | --- |
| Jméno a příjmení: | Jan Musil |
| Školní login: | Musilja2 |
| Studijní obor: | Ai3-p |
| Akademický rok: | 2022/2023 |
| Cvičení: | út 09:55 |

**Souhrnné informace**

|  |  |
| --- | --- |
| Název projektu: | Mikroblog |
| Anotace projektu: | Lehká webová aplikace, která umožňuje uživatelům vytvářet, prohlížet a mazat krátké textové příspěvky, podobně jako na Twitteru. Uživatelé se mohou zaregistrovat a přihlásit, aby mohli spravovat své příspěvky. Aplikace je vytvořená pomocí Reactu, tRPC, backendu express a MongoDB s mongoose. |

## Uživatelská dokumentace

Popište používání aplikace.

|  |
| --- |
| 1. Registrace: Chcete-li vytvořit účet, přejděte na registrační stránku a zadejte požadované uživatelské jméno a heslo. Po úspěšné registraci budete automaticky přihlášeni. 2. Přihlásit se: Pokud již máte účet, přejděte na přihlašovací stránku a zadejte své uživatelské jméno a heslo. Po přihlášení budete přesměrováni na hlavní stránku se všemi příspěvky. 3. Zobrazení příspěvků: Procházejte hlavní stránku, abyste viděli všechny příspěvky vytvořené registrovanými uživateli. Každý příspěvek zobrazuje název, obsah a uživatelské jméno autora. 4. Vytvoření příspěvku (pouze ověření uživatelé): Zadejte název a obsah příspěvku a poté klikněte na „Odeslat“. Váš nový příspěvek se objeví na hlavní stránce. 5. Smazání příspěvku (pouze ověření uživatelé a autor příspěvku): Chcete-li smazat příspěvek, který jste vytvořili, vyhledejte příspěvek na hlavní stránce a klikněte na tlačítko "Smazat" vedle něj. Příspěvek bude odstraněn ze seznamu příspěvků. 6. Odhlásit se: Pro odhlášení klikněte na tlačítko "Odhlásit". Tím vymažete váš ověřovací token a přesměrujete vás na přihlašovací stránku. |

## Dokumentace architektury

Popište klíčové koncepty a principy, na základě kterých byla navržena architektura aplikace. Popište logiku fungování aplikace.

|  |
| --- |
| Aplikace je webová aplikace založená na Reactu využívající tRPC pro komunikaci klient-server. Dodržuje zásady, jako je modularita, oddělení odpovědností a komponentový přístup. Aplikace umožňuje autorizovaným uživatelům provádět operace CRUD. Aplikační logika zahrnuje vykreslování komponent uživatelského rozhraní pro zobrazování a správu příspěvků, zpracování uživatelských interakcí a odesílání požadavků na server pro načítání dat nebo manipulaci. Události, jako je kliknutí na tlačítko, spouštějí konkrétní akce a aplikace podle toho reaguje na odpovědi serveru, jako je aktualizace mezipaměti na straně klienta nebo zobrazování chybových zpráv. |

## Dokumentace API

Popište koncové body aplikace a jejich obsluhu.

|  |
| --- |
| API poskytuje koncové body pro správu příspěvků a uživatelů. Podporuje vytváření, načítání a mazání příspěvků, registraci uživatele, přihlašování, odhlašování a získávání uživatelských informací.  Rozhraní API používá ověřování pomocí webových tokenů JSON (JWT). Když se uživatel přihlásí, obdrží JWT, který je uložen jako cookie. Token je odeslán s každým požadavkem chráněným koncovým bodům a server jej ověří, aby ověřil uživatele. Probíhají autorizace, aby bylo zajištěno, že uživatelé mohou vytvářet nebo mazat pouze své vlastní příspěvky.  Endpointy a procedury:   1. Příspěvky    * getPosts: (dotaz) Načte všechny příspěvky, včetně jejich uživatelských informací. Není vyžadováno ověření.    * createPosts: (mutace) Vytvoří nový příspěvek. Vyžaduje ověření a přijímá vstupní objekt s vlastnostmi title a content.    * deletePost: (mutace) Smaže příspěvek podle ID. Vyžaduje ověření a autorizaci uživatele (uživatel může mazat pouze své vlastní příspěvky). 2. Uživatelé    * register: (mutace) Registruje nového uživatele. Přijímá vstupní objekt s uživatelským jménem a heslem.    * login: (mutace) Přihlásí uživatele. Přijímá vstupní objekt s uživatelským jménem a heslem. Při úspěšném přihlášení vrátí JWT.    * logout: (mutace) Odhlásí uživatele vymazáním cookie autentizačního tokenu.    * getCurrentUser: (dotaz) Načte informace pro aktuálně přihlášeného uživatele. Pokud není přihlášen žádný uživatel, vrátí hodnotu null.    * getUsers: (dotaz) Načte všechny uživatele. Není vyžadováno ověření.   V případě jakýchkoli problémů rozhraní API vrací chybové zprávy. Mezi běžné chyby patří selhání ověření, chyby autorizace (např. pokus smazat příspěvek někoho jiného) a chyby nenalezen zdroj (např. pokus smazat neexistující příspěvek). Tyto chybové zprávy jsou vráceny jako prostý text s příslušnými popisy chyb, aby je klient mohl náležitě zpracovat. |

## Dokumentace databáze

Popište model jednotlivých zdrojů a vztahy mezi nimi.

|  |
| --- |
| Post model:  title (String): Název příspěvku.  content (String): Obsah příspěvku.  user (Ref<User>): Odkaz na model uživatele představující autora příspěvku. Jedná se o cizí klíč propojující model Post s modelem Uživatel.  Model Post má navíc vestavěná pole časových razítek (createdAt a updatedAt) kvůli možnosti timestamps ve schématu.  User model:  uživatelské jméno (String): Uživatelské jméno uživatele. Toto pole je povinné.  hashedPassword (String): Hašované heslo uživatele.Toto pole je povinné.  Podobně jako u modelu Post má model User také vestavěná pole časových razítek (createdAt a updatedAt) kvůli možnosti timestamps ve schématu.  Vztahy:  One-to-Many: Uživatel může mít více příspěvků, ale příspěvek může patřit pouze jednomu uživateli. Tento vztah je reprezentován uživatelským polem v modelu Post, které je odkazem na model User. |

## Kontrolní checklist

Tento protokol semestrálního projektu má obsahovat:

* Uživatelská dokumentace
* Dokumentace architektury
* Dokumentace API
* Dokumentace databáze