Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Webová aplikace pro prohlížení komiksů

2024

Adam Ryšavý

Vedoucí práce:
RNDr. Martin Trnečka, Ph.D.

Studijní program: Informační technologie,
prezenční forma

Bibliografické údaje

Autor:	Adam Ryšavý
Název práce:	Webová aplikace pro prohlížení komiksů
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2024
Studijní program:	Informační technologie, prezenční forma
Vedoucí práce:	RNDr. Martin Trnečka, Ph.D.
Počet stran:	39
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Adam Ryšavý
Title:	A web application for comics browsing
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2024
Study program:	Information Technologies, full-time form
Supervisor:	RNDr. Martin Trnečka, Ph.D.
Page count:	39
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Práce se zaměřuje na vývoj webové aplikace pro prohlížení komiksů, využívající moderní technologie jako Next.js, Express a MongoDB. Aplikace je určena jak pro běžné, tak pro registrované uživatele. Běžní uživatelé mají možnost číst komiksy v různých režimech podle jejich typu. Funkcionality usnadňující výběr komiksů jsou dostupné pouze pro registrované uživatele, zatímco editorům je poskytnuto jednoduché rozhraní pro správu komiksů. Práce je rozdělena do teoretické části, která se zabývá popisem použitých technologií, a praktické části, kde jsou detailně popsány jednotlivé části aplikace a jejich funkcionality.

Synopsis

The thesis focuses on the development of a web application for browsing comics, utilizing modern technologies such as Next.js, Express, and MongoDB. The application is designed for both regular and registered users. Regular users have the option to read comics in various modes according to their type. Features facilitating the selection of comics are available only to registered users, while editors are provided with a simple interface for managing comics. The thesis is divided into a theoretical part, which describes the technologies used, and a practical part, which provides detailed descriptions of individual parts of the application and their functionalities.

Klíčová slova: komiks; webová aplikace; Next.js; MongoDB; registrovaný uživatel

Keywords: comic; web application; Next.js; MongoDB; registered user

Děkuji vedoucímu své bakalářské práce panu RNDr. Martinu Trnečkovi, Ph.D. za jeho rady při vývoji aplikace a jejím následném designu.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

Seznam obrázků

Seznam zdrojových kódů

1 Úvod

Tradiční distribuce komiksových publikací skrze kamenné prodejny podléhala omezením týkajících se dostupnosti a fyzického uskladnění. Avšak s postupujícím vývojem digitálních technologií se komiksy staly dostupnými i ve formátu elektronických dokumentů. Tento posun otevřel nové možnosti v distribuci a autoři začali preferovat digitální publikaci nad tradičními tištěnými formáty, a to zejména kvůli širšímu dosahu a možnosti interakce se čtenáři.

Začátek distribuce komiksů ve formátu PDF představoval první krok směrem k digitalizaci. Tato forma ale přinášela určitá omezení, především z hlediska interaktivních možností a přizpůsobitelnosti. V důsledku toho se rozvinula poptávka po specializovaných webových platformách, které by umožnily uživatelům prohlížet komiksy online s větší flexibilitou a rozšířenými funkcemi.

Jakožto komiksový čtenář jsem nespokojen s vývojem existujících komiksových webových stránek. Většina z nich používá jednotnou šablonu systému WordPress, což vede ke stejným opakujícím se nedostatkům. Proto jsem se rozhodl vytvořit vlastní webovou aplikaci, která tyto nedostatky řeší a nabízí moderní technologická řešení. Mým cílem je vytvořit aplikaci, která bude mít vysokou škálovatelnost a bude schopna uspokojit potřeby komiksových nadšenců v dlouhodobém horizontu.

Aplikace je navržena pro dva typy uživatelů: běžné a registrované. Běžní uživatelé mají přístup ke všem komiksům a jejich kapitolám, avšak jsou vystaveni občasným vyskakovacím reklamním oknům. Naopak registrovaní uživatelé mají přístup ke všem funkcím aplikace, jako je možnost komentování, hodnocení a další, a to bez zobrazování jakýchkoliv reklamních bannerů.

2 Použité technologie a balíčky

Při vývoji aplikace bylo zapotřebí zkombinovat mnoho technologií. Zde je popíšu a zároveň vysvětlím, jak je kombinuji.

2.1 HTML

HTML [1] je značkovací jazyk, který slouží k vytváření a strukturování obsahu webových stránek. Byl vyvinut v roce 1990 Timem Bernersem Leem s cílem usnadnit sdílení informací na internetu a od té doby prošel několika změnami. V HTML se obsah stránky dělí do souvisejících bloků pomocí značek, nazývaných také elementy. Tyto značky se obvykle zapisují mezi znaky < a >, které ohraničují daný element a jsou doplněny jeho názvem, jako například `<div>` nebo `<span>`.

2.2 CSS

CSS [2, 3] je technologie pro vizualizace webové stránky. Její zápis může být umístěn buď do samostatného souboru s příponou `.css`, nebo přímo v rámci elementu pomocí atribut `style`. V CSS vizualizujeme prvky pomocí CSS pravidel, které mohou být založeny na druhu elementu (například `p`) nebo na konkrétní třídě či identifikátoru. Následně v rámci složených závorek definujeme vlastnosti ve formátu `název:hodnota` . Dědičnost CSS zajišťuje, že některé vlastnosti aplikované na rodičovský prvek se projeví i na všechny jeho vnořené prvky (potomky).

2.3 JavaScript

Programovací jazyk JavaScript [4] zajišťuje interaktivní a funkční aspekty webových stránek. Jeho využití však nekončí pouze na úpravách v prostředí webového prohlížeče, ale nalézá uplatnění i při vývoji webových aplikací. JavaScript umožňuje dynamickou manipulaci s obsahem stránky jako reakci na uživatelské akce. Skrze události, které uživatel provádí, jsme schopni implementovat logické operace a reakce, což nám umožňuje modifikovat chování různých prvků stránky v souladu s uživatelským vstupem. Jazyk JavaScript je implementován buď pomocí elementu `<script>`, nebo jako externí soubor s příponou `.js`, který je do stránky načten.

2.4 Node.js

Node.js [5] je open-source běhové prostředí pro vývoj aplikací, které umožňuje použití jazyka JavaScript pro tvorbu backendové části aplikací. Tím rozšiřuje možnosti využití JavaScriptu, který je tradičně spojen s frontendem webových stránek na tvorbu backendu. Node.js disponuje širokým výběrem knihoven, které usnadňují vývoj a poskytují rozšířenou funkcionalitu pro tvorbu webových aplikací.

2.4.1 Npm

Node.js využívá správce balíčků npm (Node Package Manager) [6] k instalaci a správě knihoven. Všechna metadata o těchto balíčcích jsou uložena v souboru `package-lock.json`. Npm umožňuje uživatelům nejen provádět instalaci balíčků, ale také aktualizovat stažené balíčky a spravovat jejich verze. Další poskytnutá funkce je detekce a řešení konfliktů, které vznikají v důsledku použití různých verzí knihoven.

2.5 Bootstrap

Bootstrap [7] je frontendový framework zaměřený na vytváření responzivních webových stránek. Jeho použití umožňuje snadnou implementaci responzivity a přizpůsobení obsahu stránky různým zařízením a velikostem obrazovek. Framework nabízí předdefinované CSS třídy a komponenty, které umožňují rychlé a efektivní designování webových rozhraní. Bootstrap lze integrovat do projektu buď přímým připojením pomocí odkazu v HTML kódu stránky, nebo instalací balíčku pro určité frameworky, jako je například React.

2.6 React

React [8] je JavaScriptová knihovna pro tvorbu uživatelského rozhraní. Využívám dvě hlavní části této knihovny.

2.6.1 Komponenty

Komponenty umožňují strukturovat kód do souvislých celků. Každá komponenta obsahuje kombinaci JavaScriptového kódu, HTML a je doplněna o vlastní soubor s kaskádovými styly `module.css`, který definuje vzhled komponenty. Kombinací více komponent vytvářím výsledné webové stránky.

2.6.2 Hooky

Hooky umožňují manipulovat se stavem komponenty. V následujícím seznamu uvedu ty, které jsem v aplikaci nejčastěji použil.

- **`useEffect`**: Tento hook používám k inicializaci stavu komponenty a provádění vedlejších efektů, jako je získávání dat z serveru pomocí HTTP požadavků.
- **`useState`**: Pomocí tohoto hooku uchovávám stav vnitřních proměnných komponenty, které následně zobrazuji v HTML.
- **`UseRef`**: Tento hook umožňuje referovat na konkrétní část komponenty.

2.6.3 next.js

Next.js [10] je framework postavený na Reactu, který slouží k tvorbě moderních webových aplikací. Tento framework poskytuje širokou škálu optimalizací a funkcí pro vývojáře. Jeho klíčovou vlastností je schopnost renderovat stránky na straně serveru, což přináší výhody v oblasti výkonu a optimalizace pro prohlížeče.

2.7 react-bootstrap

React-bootstrap [11] rozšiřuje možnosti vizualizace přidáním speciálních komponent s předdefinovanými vlastnostmi. V projektu používám zejména `<Row>` a `<Col>`, které definují strukturu gridu. Pomocí gridu pak upravuji rozložení jednotlivých komponent.

2.7.1 react-bootstrap-icons

React-bootstrap-icons [12] poskytuje sadu ikon, které mohou být použity jako samostatné komponenty ve frameworku Bootstrap.

2.8 axios

Axios [13] je balíček pro vytváření HTTP požadavků, který používám pro komunikaci s backend serverem. Pomocí těchto požadavků synchronizuji data uživatelů s databází a aktualizuji stav komponenty.

2.9 express

Express [14] je webový framework pro Node.js, který poskytuje sadu funkcí pro vytváření a zpracování webových aplikací a API. V projektu je Express využíván pro zpracování HTTP požadavků, které přicházejí od klientů. Pomocí metody `listen` je server konfigurován k naslouchání na určeném portu, na kterém přijímá příchozí komunikaci. Na základě URL adresy Express rozhoduje o tom, jaká cílová adresa bude použita k zpracování daného požadavku.

2.10 cors

Balíček cors [15, 16] (Cross-Origin Resource Sharing) poskytuje mechanismus, který umožňuje webovým aplikacím komunikovat a sdílet zdroje mezi různými doménami. Standardní bezpečnostní politika prohlížečů, známá jako Same Origin Policy, obvykle omezuje takový přístup. V projektu je tento balíček použit pro nastavení povolení přístupu ke zdrojům z adresy frontend serveru, jak je demonstrováno následujícím kódem 1.

```
app.use(cors({
  origin: "http://localhost:3000",
  credentials: true
}));
```

Zdrojový kód 1: Cors v backend serveru

2.11 MongoDB

MongoDB [9] je open source NoSQL databáze, která se od tradičních SQL relačních databází odlišuje tím, že nepracuje s tabulkami, ale s dokumenty. Dokumenty jsou strukturovány ve formátu klíč-hodnota a využívají notaci podobnou JSON, známou jako BSON. Jedním z klíčových prvků MongoDB je schopnost zanořování dokumentů, což umožňuje vytvářet složitější datové struktury. Každý dokument v kolekci má primární klíč nazývaný identifikátor dokumentu (id). Kolekce sdružují soubory dokumentů s podobnou strukturou.

2.11.1 mongoose

Balíček mongoose [17] je objektový modelovací nástroj pro MongoDB. Pomocí Mongoose můžeme definovat schémata objektů, která nám umožňují specifikovat strukturu dat ukládaných do databáze. V mém projektu jsou definována hlavní schémata pro komiksy, uživatele a komentáře. Tato schémata určují formát a validaci dat, která jsou ukládána do databáze a zjednodušují manipulaci s nimi v rámci aplikace.

2.12 bcrypt

Balíček bcrypt [18] je využíván pro hashování hesel v aplikaci. Při registraci uživatele je použita hashovací funkce bcryptu k vytvoření hashe hesla, který je následně uložen v databázi. Při přihlašování uživatele se porovnává uložený hash v databázi s hashem získaným z hesla zadaného uživatelem. Bcrypt také poskytuje možnost přidání soli k hashi hesla, což zvyšuje bezpečnost proti slovníkovým útokům.

2.13 jsonwebtoken

Jsonwebtoken [19] slouží k vytváření a ověřování JSON Web Tokenů (JWT), které jsou často používány pro autentizaci uživatelů webových aplikacích. Po úspěšném ověření uživatelského jména a hesla v procesu přihlašování je vytvořen JWT, který je uložen do `session storage` prohlížeče. Tento token obsahuje zahashovaný klíč, který je generován pomocí funkce `crypto`, a je mu nastavena životnost. Kromě toho je do tokenu zakódováno uživatelské id, jméno a příznak editora, pokud má uživatel odpovídající oprávnění.

2.14 nodemon

Nodemon [20] je utilita pro sledování změn v kódu Node.js aplikací. Její hlavní funkcí je automatické restartování serveru v reakci na detekované změny v kódu. Tento proces umožňuje vývojářům pracovat na serverové části aplikace a okamžitě vidět změny provedené ve zdrojovém kódu bez nutnosti manuálního restartování serveru. Nodemon usnadňuje iterativní vývoj a zvyšuje produktivitu vývojářů.

2.15 nodemailer

Balíček nodemailer [21] umožňuje odesílat emaily. Pomocí něj můžeme uživateli zaslat token k resetu hesla.

2.16 dotenv

Balíček dotenv [22] umožňuje načítat konfigurační proměnné pro různá prostředí aplikace. Jeho funkce spočívá v poskytnutí souboru `.env`, který slouží k uchování těchto proměnných. Pomocí objektu `process.env` jsou následně tyto proměnné načítány do běhového prostředí aplikace, což k nim umožňuje přístup v kódu.

2.17 Nahrávání souborů

Zde popíšu technologie použité pro nahrávání souborů.

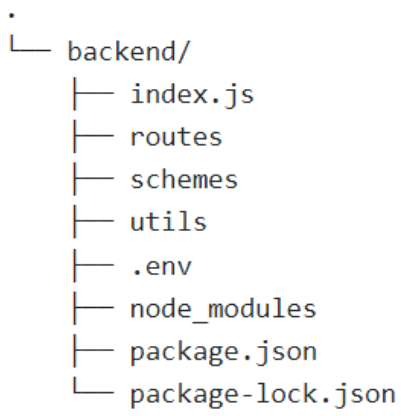
- **react-dropzone:** React-dropzone [23] je balíček poskytující komponentu, která umožňuje uživatelům nahrávat soubory na webové stránky přetažením těchto souborů do oblasti komponenty.
- **adm-zip:** Tento balíček [24] slouží k extrahování souborů `.zip`.
- **multer:** Balíček [25] slouží k zpracování a nahrávání souborů pomocí formulářů `multipart/form-data`.

3 Programátorská dokumentace

Aplikace je rozdělena na dva servery. Backendový server je zodpovědný za zpracování HTTP požadavků od klienta, transformaci dat a interakci s databází. Frontendový server je určen k renderování komponent na základě dat z požadavků a následnému odesílání těchto komponent klientovi. V této sekci vystvětlím jednotlivé části těchto serverů.

3.1 Struktura backend serveru

Tento obrázek 1 ilustruje strukturu backendového serveru, který je součástí aplikace. Backendový server, stejně jako frontendový server, je založen na platformě Node.js.



Obrázek 1: Struktura backend serveru

3.2 mongoose schémata

Aplikace zahrnuje šest schémat a jeden modelový soubor, ve kterém jsou ze schémat vytvářeny a exportovány modely. Všechny soubory jsou umístěny ve složce `schemes`.

3.2.1 schéma komiksu

Schéma pro definování komiksů 2. Každý komiks o sobě nese základní informace, id komentářů příslušících komiksu a kapitoly definované pomocí schématu kapitoly.

```
const mongoose = require('mongoose');
const chapterSchema = require('./chapterSchema');
const comicSchema = new mongoose.Schema({
  comicId: mongoose.ObjectId,
  title: String,
  type: String,
  author: String,
  summary: String,
  released: Date,
  updated: Date,
  genres: [String],
  rating: Number,
  rated: Number,
  readingStyle: String,
  chapters: [{
    type: Map,
    of: chapterSchema
  }],
  status: String,
  comments: [{
    type: mongoose.Schema.Types.ObjectId,
    ref: 'comments',
  }]
});
module.exports = comicSchema;
```

Zdrojový kód 2: schéma komiksu

3.2.2 schéma kapitoly

Schéma kapitoly 3 obsahuje datum přidání, číslo kapitoly a počet částí (obrázků).

```
const mongoose = require('mongoose');
const chapterSchema = new mongoose.Schema({
  date: Date,
  chapterNum: Number,
  parts: Number
});
module.exports = chapterSchema;
```

Zdrojový kód 3: schéma kapitoly

3.2.3 schéma uživatele

Schéma uživatele 4 obsahuje jeho základní informace, pole s oblíbenými komiksy, pole hodnocených komiksů, pole rozečtených komiksů a pole komentářů, kterým dal „líbí se“ nebo „nelíbí se“.

```
const mongoose = require('mongoose');

const usersSchema = new mongoose.Schema({
  username: String,
  email: String,
  password: String,
  gender: String,
  editor: Boolean,
  favorite: [String],
  rated: [{
    title: String,
    rating: Number
  }],
  stoppedReading: [{
    title: String,
    chapter: Number
  }],
  commentsLiked: [{
    type: mongoose.Schema.Types.ObjectId,
    ref: 'comments'
  }],
  commentsDisliked: [{
    type: mongoose.Schema.Types.ObjectId,
    ref: 'comments'
  }],
});

module.exports = usersSchema;
```

Zdrojový kód 4: schéma uživatele

3.2.4 schéma komentáře

Schéma komentáře 5 se skládá z id komiku, uživatele, datumu, textu komentáře, počtu „líbí se“, „nelíbí se“ a pole id podkomentářů.

```
const mongoose = require('mongoose');

const commentSchema = new mongoose.Schema({
  comicId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'comics',
  },
  user: String,
  text: String,
  timestamp: {
    type: Date,
    default: Date.now,
  },
  like: Number,
  dislike: Number,
  subComments: [{
    type: mongoose.Schema.Types.ObjectId,
    ref: 'subcomments',
  }],
});

module.exports = commentSchema;
```

Zdrojový kód 5: schéma komentáře

3.2.5 schéma podkomentáře

Schéma podkomentáře 6 je podobné jako u komentáře, navíc obsahuje id rodiče.

```
const subCommentSchema = new mongoose.Schema({
  user: String,
  text: String,
  timestamp: {
    type: Date,
    default: Date.now,
  },
  like: Number,
  dislike: Number,
  repliedTo: String,
  parentComment: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'comments',
  },
});
module.exports = subCommentSchema;
```

Zdrojový kód 6: schéma podkomentáře

3.2.6 schéma resetovacího tokenu

Schéma tokenu 7 má id uživatele, náhodný řetězec bytů a expirační dobu.

```
const resetTokenSchema = new mongoose.Schema({
  userId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
  },
  token: {
    type: String,
  },
  createdAt: {
    type: Date,
    default: Date.now,
    expires: '1h'
  }
}, { collection: 'resetTokens' });
module.exports = resetTokenSchema;
```

Zdrojový kód 7: schéma resetovacího tokenu

3.3 utils

V adresáři `utils` je umístěn mechanismus tokenové autentizace. Při zpracování každé routy, která vyžaduje ověření autentičnosti, je tato funkce vyvolána jako první. V rámci této funkce je využíván klíč uložený v souboru `.env` k dekódování tokenu.

3.4 routes

Routy jsem umístil do složky `routes` a pro lepší přehlednost jsem pro každou z nich vytvořil samostatný soubor. Všechny tyto soubory jsou následně importovány do souboru `index.js` pro použití. Nyní se zaměřím na jejich popis a zároveň je rozdělím podle obsahu se kterým pracují.

3.4.1 komiksové routy

Tyto routy pracují převážně s komiksovým schématem. Jsou zodpovědné za úpravu informací o komiksu, nebo jejich poskytování.

- **getSetOfTwentyComics:** Tato routa poskytuje sadu dvaceti komiksů. Jako argument přijímá číslo, kde hodnota jedna znamená dvacet nejnovějších komiksů. Další čísla postupně poskytují starší komiksy.
- **search:** V této routě obdržíme název komiksu, jeho typ a žánry. Na základě těchto informací vytváříme `regulární výrazy`, které používáme k vyhledání odpovídajících položek v databázi. Následně vracíme sadu komiksů, které splňují všechny zadané požadavky.
- **getComic:** Dodá jeden komiks podle jeho jména.
- **slideshowInfo:** Poskytne čtyři nejlépe hodnocené komiksy.
- **getChapter:** Tato routa poskytuje jednotlivé kapitoly komiksu podle jejich čísla.
- **getComicTitles:** Routa dodává seznam titulů pro editor.
- **saveComic:** Routa ukládá komiks z editoru.
- **uploadComic:** Tato routa nahrává nové komiksy z externích souborů v editoru.
- **readingInfo:** Poskytuje informace o způsobu čtení komiksu.

3.4.2 uživatelské routy

Tyto routy aktualizují informace o uživatelích a komiksech. Pracují se schématy komiksu a uživatele.

- **registration:** Tato routa provádí registraci nových uživatelů. Nejdříve se dotazuje databáze, zda již existuje uživatel se zadaným uživatelským jménem nebo emailem. Pokud uživatel existuje, je vrácena odpovídající zpráva. V opačném případě pokračuje registrace šifrováním hesla pomocí funkce `bcrypt` a vytvořením nového profilu s zašifrovaným heslem.
- **login:** Tato routa je zodpovědná za proces přihlašování uživatele. Porovnává hashované heslo, které bylo zasláno, s heslem v profilu uživatele. V případě shody je uživateli odeslán „autentizační token“, který obsahuje jeho oprávnění. Pokud uživatel nebyl nalezen, nebo se hesla neshodují, jsou vráceny odpovídající zprávy.
- **checkFavorite:** Routa zkontroluje, zda má uživatel komiks v oblíbených.
- **addFavorite:** Routa obdrží informace o komiksu a přidá jej do seznamu oblíbených uživatele. Pokud se stejný záznam již v seznamu uživatele nachází, je z něj odebrán.
- **probeRating:** Routa zjišťuje, zda uživatel již hodnotil komiks.
- **rating:** Pomocí této routy jsou komiksy hodnoceny. Pokud uživatel již daný komiks hodnotil, je jeho hodnocení odečteno od celkového součtu hodnocení komiksu a sníží se počet recenzentů.
- **getProfile:** Dodává profil uživatele.
- **stoppedReading:** Tato routa aktualizuje záznamy rozečtených komiksů uživatele.
- **userReadChapters:** Tato routa naopak vrací záznamy o konkrétním rozečteném komiksu uživatele.

3.4.3 komentářové routy

Tyto routy obsluhují komentáře u jednotlivých komiksů a reakce na ně.

- **postComment:** Tato routa zajišťuje přidání komentáře k danému komiksu. Rozlišuje mezi běžným komentářem a odpovědí (podkomentářem) na základě přítomnosti id rodiče. V případě podkomentáře je nastaveno id nového komentáře i do jeho rodiče.

- **like/dislikeComments:** Dvě podobné routy se zaměřením na přidání nebo odebrání určitého stavu. Každá je zodpovědná za jednu operaci a jsou odděleny kvůli vzájemné závislosti. Když uživatel změní „líbí se“ na „nelíbí se“, je nejprve zavolána funkce pro odebrání „líbí se“ a až poté funkce pro přidání „nelíbí se“.
- **fetchComments:** Routa poskytuje množinu deseti komentářů k danému komiksu. Jejich stáří se určuje obdobně jako u routy „getSetOfTwentyComics“, a to číslem z argumentu. Využíváme zde funkci `populate`, která vymění id komentářů v komiksu za příslušné dokumenty.
- **getTopComment:** Dodává nejlépe hodnocený komentář k danému komiksu.
- **commentReactions:** Poskytuje informace o reakcích „líbí se“ a „nelíbí se“ u komentářů, na které uživatele reagoval.

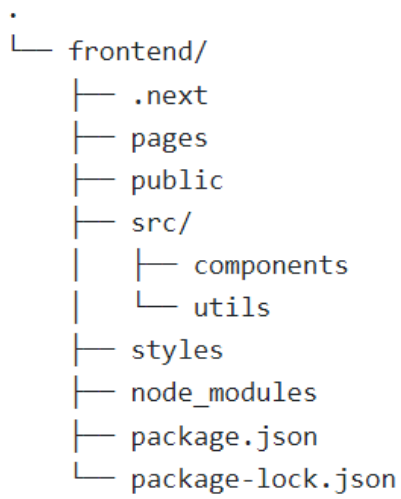
3.4.4 resetPassword routa

Tato routa je určena pro proces obnovení hesla a jako jediná pracuje se schématem `resetToken`. Skládá se ze dvou fází:

- **Žádost a email.** Nejprve routa vygeneruje „resetovací token“, který je přiřazen konkrétnímu uživateli. Tento token je uložen do časované kolekce v databázi a zároveň je zaslán klientovi emailem.
- **Validace tokenu a reset.** Poté co klient obdrží token z emailu a odešle jej zpět s novým heslem, je ověřena jeho platnost. Pokud je platný, nastaví se nové heslo uživateli.

3.5 frontend server

Následující obrázek 2 zachycuje strukturu frontend serveru, který je založený na Reactu a Next.js



Obrázek 2: Struktura frontend serveru

3.6 src/components

V této složce se nacházejí komponenty aplikace. Každá komponenta je tvořena dvěma soubory: `[název].js` a `[název].module.css`. Hlavní výhoda použití modulárního přístupu spočívá v tom, že se minimalizuje přenos nevyužitého kódu při načítání stránky. Tím se zlepšuje výkon a škálovatelnost aplikace.

3.7 src/utils

V adresáři `utils` jsou umístěny soubory obsahující často používané funkce, jako například ověřování přítomnosti „autentizačního tokenu“ a extrakce dat z něj.

3.8 public

V tomto adresáři jsou uloženy veškeré obrázky aplikace. Kapitoly se nacházejí ve složce `comics` a jsou děleny podle názvu komiksu a čísla kapitoly. Čísla částí kapitol jsou doplněna nulami na začátku, aby dosáhla čtyřmístného formátu a jsou uloženy ve formátu `.jpg`.

3.9 pages

Next.js zajišťuje routování prostřednictvím souborů ve složce `pages`. Každý soubor reprezentuje samostatnou komponentu a na základě jeho jména je automa-

ticky vytvořena odpovídající stránka. Stránky mohou být také vnořené v podadresářích, což umožňuje vytvářet hierarchické routy. Další možností je vytvářet dynamické routy pomocí závorek `[]`. Nyní detailně popíšu jednotlivé stránky a použité komponenty. Některé komponenty budou popsány samostatně kvůli jejich důležitosti nebo použití v jiných komponentách. Standardní komponenty použité na každé stránce jsou „Header“ a „Footer“.

- **Header:** Header je komponenta, která tvoří záhlaví každé stránky a je rozdělena do tří částí. První část obsahuje „ikonku“ aplikace, která umožňuje uživatelům navigovat zpět na hlavní stránku. Druhou částí je komponenta „searchBar“, která umožňuje uživatelům vyhledávat komiksy na stránce. Po aktivaci se zobrazí vysouvací menu z vrchní části stránky, obsahující kategorie vyhledávání, jako je název komiksu, jeho typ a jednotlivé žánry. Po kliknutí na tlačítko vyhledat je uživatel přesměrován na stránku „/search“ a parametry vyhledávání jsou ukládány do URL adresy. Poslední částí je „menu“, které umožňuje uživatelům navigovat na další stránky aplikace. Některé odkazy jsou přístupné pouze uživatelům s odpovídajícími oprávněními v rámci „autentizačního tokenu“.
- **Footer:** Footer je komponenta umístěná v zápatí každé stránky a obsahuje pouze „ikonku“ stránky, která je zarovnána na střed. V případě potřeby lze do tohoto zápatí přidat kontaktní informace překladatelů nebo jiné relevantní údaje.

3.9.1 index

Toto je hlavní stránka aplikace, která se skládá ze dvou hlavních komponent. První komponentou je „Slideshow“, která slouží k zobrazení komiksů s nejlepším hodnocením. Tato komponenta je založena na komponentě „Carousel“ z frameworku Bootstrap. Při zobrazení na mobilních zařízeních a tabletech zobrazuje titulní obrázky komiksů, zatímco při zobrazení na počítačích je k těmto obrázkům přidán také název a popis příslušného komiksu. Každý obrázek v Slideshow může po kliknutí přesměrovat uživatele na stránku daného komiksu. Druhou hlavní komponentou je „ComicCardBrowser“, která přijímá pole komiksů jako argument. Tato komponenta rozloží pole na jednotlivé komiksy a pomocí nich generuje kartičky komiksů. V závislosti na šířce obrazovky rozhoduje, kolik kartiček se vedle sebe uspořádá. Vzhled a uspořádání jednotlivých kartiček řídí komponenta „ComicCard“. Komponenta „ComicCardBrowser“ má nastaven maximální počet najednou zobrazených komiksů na dvacet. Pokud se nachází na úvodní stránce, je zobrazeno tlačítko „Browse more“, které uživatele přesměruje na stránku „/comics“, kde lze zobrazit další komiksy.

3.9.2 /comics

Tato složka obsahuje stránky umožňující výběr ze všech komiksů, zobrazení podrobných informací o jednotlivých komiksech a jejich čtení v rámci kapitol.

- **index:** Tato stránka slouží k prohlížení komiksů. Stejně jako na úvodní stránce se zde využívá komponenta „ComicCardBrowser“, avšak bez omezení na maximální počet zobrazených komiksů. V případě, že v databázi existuje více než dvacet komiksů, se na spodní části stránky zobrazí řada tlačítek. Každé tlačítko umožňuje přepnout na další sadu dvaceti komiksů pomocí požadavku na routu `getSetOfTwentyComics` a následného zobrazení komiksů z odpovědi. Zároveň je maximálně zobrazeno pět tlačítek, přičemž jejich číslo se může lišit o pouze dvě od vybraného. Následující kód 8 ukazuje proces tvorby těchto přepínačů a výpočet jejich počtu pro zobrazení.

```
1        const numButtons = Math.ceil(comicCount / 20);
2
3        const startButton = Math.max(1, Math.min(comicSet - 2,
             numButtons - 4));
4
5        const endButton = Math.min(numButtons, startButton + 4);
6        var buttons = [];
7        for (let i = startButton; i <= endButton; i++) {
8         buttons.push(
9         <button key={i} className={` \${styles["switching-button"]}`
              } onClick={() => {
10             setComicSet(i);
11             fetchComics(i);
12          }}{i}> </button>
13          )};
```

Zdrojový kód 8: Tvoření přepínacích tlačítek

- [**title**]**:** Po rozkliknutí komiksu se spustí funkce `getServerSideProps`, která provede požadavek na získání informací o daném komiksu ještě před odesláním komponenty klientovi. Hlavní komponentou této stránky je „ComicOpen“, která přijímá komiks jako argument a předává ho dalším komponentám. V této komponentě jsou zobrazeny všechny podrobnosti o komiksu. „ComicOpen“ také využívá komponenty, které jsou určeny pouze pro registrované uživatele a omezuje k nim přístup běžným uživatelům. Tyto komponenty pro registrované uživatele vždy prověřují záznamy v jejich profilu a na základě toho mění svůj počáteční stav.
  - První takovou komponentou je „Favorites“, která uživatelům umožňuje přidávat a odebírat komiksy ze seznamu oblíbených. Tato komponenta se skládá ze dvou grafických prvků pro zobrazení hvězdy, přičemž jeden prvek tvoří ohraničení druhého. Pokud má uživatel komiks již přidaný mezi oblíbenými, tak komponenta umožní jeho odebrání.
  - Další komponentou pro registrované uživatele je „Rating“, která umožňuje uživatelům hodnotit komiksy. Při interakci s touto komponen-

tou se otevře kontejner obsahující pět klikatelných hvězd. Po kliknutí na hvězdu se změní hodnota proměnné „setRating“ na index kliknuté hvězdy, která způsobí změnu barvy hvězd podle tohoto indexu a všech předchozích. Během procesu hodnocení je aktivována funkce `handleClickOutside`, která sleduje, zda uživatel klikl mimo kontejner, pokud ano, skryje kontejner a obnoví hodnocení na původní stav. Po dokončení hodnocení je nastavena proměnná `isRating`, která zobrazí odeslané hodnocení. Pokud uživatel již komiks hodnotil, komponenta umožní odebrání hodnocení. V případě, že uživatel není přihlášen, je tato komponenta skryta.

– Poslední komponentou je „CommentSection“, kterou popíšu v samostatné podkapitole.

Společně s komponentou „Rating“ je zobrazena komponenta „ComicScore“, která prezentuje číselné hodnocení komiksu získané vydělením celkového hodnocení počtem recenzí.

Další komponentou je „ComicChaptersList“, která zobrazuje seznam kapitol komiksu. Pokud je uživatel přihlášen, provede se požadavek na získání informací o komiksech, které již uživatel začal číst. Pokud je v odpovědi záznam o čtení tohoto konkrétního komiksu, tak komponenta zobrazí upozornění u poslední čtené kapitoly.

K upozornění uživatele na omezení přístupu k určité funkcionalitě slouží komponenta „Popover“. Tato komponenta získá souřadnice cílové komponenty pomocí hooku `useRef` a zobrazí se přímo nad ní. „Popover“ se používá pouze u komponent, které jsou viditelné, ale nejsou přístupné pro běžné uživatele.

- **[title]/[chapter-title]:**

  Na této stránce jsou zobrazeny kapitoly komiksu. Nejprve se spustí funkce `getServerSideProps`, která odesílá požadavek na získání částí kapitoly a poté je klientovi zaslána. Dalším důležitým krokem je získání stylu čtení komiksu a informace o poslední kapitole. Styl čtení je předán komponentě „Reader“ a bude platný i pro další kapitoly stejného komiksu. Číslo poslední kapitoly je předáno komponentě „NavigationButtons“.

  Komponenta „Reader“ přijímá jako vstupní parametr počet částí komiksu a na základě nich vytváří pole relativních cest k obrázkům. Tyto obrázky jsou následně načteny a podle zvoleného stylu čtení je upraveno jejich rozložení. Pokud je zvolený styl čtení „vertikální“, obrázky jsou umístěny bez mezer za sebou. Uživateli je také zobrazena šipka na dolním okraji obrazovky, která mu umožní snadný návrat na začátek kapitoly. V případě „horizontálního“ stylu je použita komponenta „Carousel“, která umožňuje uživateli procházet komiks jako knihu.

K navigaci mezi kapitolami slouží komponenta „NavigationButtons“, která na konci každé kapitoly zobrazuje tlačítka vedoucí na následující nebo předchozí kapitolu. V případě, že uživatel došel na první nebo poslední kapitolu komiksu, zobrazí tlačítka, která umožňují návrat zpět na stránku s přehledem komiksu.

Aplikace také obsahuje komponentu „Advertisement“, avšak z důvodu usnadnění testování není využívána. Při reálném nasazení by běžní uživatelé museli zavřít reklamní okno, aby mohli pokračovat ve čtení.

3.9.3 CommentSection

Tato komponenta spravuje komentáře k jednotlivým komiksům. Nejprve je odeslán požadavek na získání deseti nejnovějších komentářů a jejich odpovědí, a pokud je uživatel přihlášen, tak také požadavek na jeho reakce k těmto komentářům. Komentáře jsou následně uloženy do pole pomocí funkce `setComments` a jsou vypsány v hlavní sekci. Reakce uživatelů jsou ukládány pomocí funkcí `setLikes` a `setDislikes`. Komponenta se skládá z několika sekcí, které budu dále popisovat.

- **GetTopComment:** Tato komponenta se nachází na vrchní části a zobrazuje komentář s nejvyšším počtem „líbí se“.
- **Hlavní komentářová sekce:** Zde jsou vypsány všechny komentáře s jejich celkovým počtem a `textarea` pro přidání nového komentáře. Každý komentář je rozdělen do pěti částí.
  1. Autor a datum.
  2. Obsah komentáře.
  3. Komponenta `HandThumbsUp` pro „líbí se“.
  4. Komponenta `HandThumbsDown` pro „nelíbí se“.
  5. Tlačítko „reply“.
- **Sekce podkomentářů:** Pokud existují odpovědi na daný komentář, je vytvořena další sekce pro zobrazení těchto podkomentářů.
- **Tlačítko More comments:** Podobně jako u komponenty „ComicCardBrowser“ je odeslán požadavek na získání dalších deseti komentářů pomocí funkce `LoadMoreComments`, které budou přidány za poslední komentář v hlavní sekci.

Nyní popíši mechanismy pro vytváření komentářů a jejich reakcí. Každý komentář, poté co je odeslán do databáze, je nezbytné zařadit mezi stávající komentáře pomocí funkce `setComments`. Hlavní komentář je normálně zařazen přímo do pole komentářů. V případě podkomentáře je nejprve nutné najít jeho rodičovský komentář pomocí identifikátoru `parentId` a poté ho do jeho podkomentářů.

Pokud uživatel chce reagovat na komentář tlačítky „líbí se“ nebo „nelíbí se“, nebo na něj odpovědět tlačítkem „reply“, zavolá se funkce `handleButtonClick`. Komentářová sekce také využívá komponentu „Popover“, a to zda se má zobrazit, je určeno právě touto funkcí. Pokud uživatel není přihlášen, jsou souřadnice tlačítka vypočteny z jeho reference a zobrazí se okno „Popover“ s upozorněním, že uživatel nemá právo provést tuto akci. Pokud je uživatel přihlášen, funkce pokračuje dále.

V případě „líbí se“ či „nelíbí se“ se nejprve kontroluje, zda už uživatel nemá jednu z reakcí u komentáře, pokud ano, je nejdříve zavolána opačná akce, která stav odebere a následně se pokračuje v původní. V obou funkcích se aktualizuje stav „líbí se“ či „nelíbí se“ u uživatele a nakonec jsou aktualizovány i komentáře pomocí funkce `UpdateCommentState`.

3.9.4 login, registration a resetPassword

Jelikož se jedná o velmi podobné stránky, rozhodl jsem se je popsat dohromady. Každá z těchto stránek obsahuje vlastní komponentu s formulářem.

- **„LoginForm“** obsahuje vstupní pole pro jméno a heslo uživatele. Po úspěšném přihlášení je uživateli do `session storage` uložen „autentizační token“ a je přesměrován na hlavní stránku. Z této stránky je také možné přejít na „registrační stránku“ pomocí tlačítka „Not registered yet? Click here!“.

- **„RegistrationForm“** formulář je rozšířený o pole pro email, pohlaví a kontrolu hesla. Email je kontrolován metodou `setCustomValidity`, která ověřuje přitomnost zavináče. Stejně tak je ověřována délka hesla, aby byla delší než sedm znaků. Před odesláním formuláře je také ověřeno, zda jsou hesla v obou polích stejná.

- **„ResetForm“** formulář na začátku obsahuje pouze pole pro zadání emailu. Poté, co uživatel zadá správný email, je formulář rozšířen o pole pro zadání „validačního tokenu“ a nového hesla. Po odeslání emailu s „validačním tokenem“ má uživatel hodinu na změnu hesla.

Všechny formuláře obsahují elementy `<div>` pro zobrazení chybných vstupů. Pokud se z backendu vrátí zpráva s chybou, je tato zpráva zobrazena s ikonou „ExclamationTriangleFill“.

3.9.5 search

Stránka search má v URL nastavené parametry pro hledání z komponenty „searchBar“. Ve funkci `fetchSearchResults` nejprve extrahujeme parametry z URL pomocí funkce `router` a ty pak pošleme s požadavkem na backend server. Komiksy z odpovědi předáme do komponenty „ComicCardBrowser“, která je následně zodpovědná za vypsání nalezených komiksů.

3.9.6 user-profile

Stránka pro výpis informací o uživateli. Hlavní komponentou je zde „UserProfile“, ve které se nejdřív funkcí `fetchProfile` odešle požadavek pro získání profilu uživatele. Následně z něj vypisujeme všechny kategorie. Pod kategoriemi je tlačítko „Log out“, které odebere „autentizační token“ ze `session storage`.

3.9.7 editor

Tato stránka umožňuje uživatelům s „příznakem editor“ upravovat a přidávat komiksy. Hlavní komponentou je zde „ComicEditor“, která je rozdělena do tří částí pomocí přepínacích tlačítek.

- **„Add new a Edit comic:“** Obě části jsou založeny na stejném formuláři. Při přidávání nového komiksu je formulář prázdný a vyžaduje vyplnění všech polí. Při úpravě existujícího komiksu je nejdříve spuštěna funkce `fetchComicTitles`, která získá seznam názvů komiksů pro editaci. Po výběru komiksu je zavolána funkce `retrieveComic`, která pošle požadavek na vybraný komiks a nastaví všechny jeho informace do formuláře.

  Na konci formuláře se nachází komponenta „ChapterDropZone“, pomocí které může editor nahrávat soubor `.zip` s kapitolami a titulním obrázkem komiksu do složky `public`. Funkce `onDrop` nejprve zkontroluje autentičnost editora pomocí požadavku na routu `get-profile`. Z profilu zkontroluje `příznak editora` a v případě úspěchu je soubor zaslán na frontend routu `api/upload-chapter`. Routa soubor rozbalí ve složce `public/comics` a v případě přítomnosti titulního obrázku jej přesune do složky `public/comic-icons`. Jako odpověď routa vrací obsah souboru získaný z funkce `readdirSync`.

- **„Upload comic:“** Zde je využita komponenta „DropZone“, která umožňuje uživatelům nahrávat komiksy ve formátu `.json`. Editor zde může stáhnout šablonu pro vyplnění, a při následném nahrání se pomocí metody `Object.keys()` kontrolují `klíče` souboru. Chybějící nebo neplatné `klíče` jsou pak uloženy do pole `missingKeys` a zobrazeny uživateli. Uživatel není oprávněn odeslat komiksy, které neprošly touto kontrolou.

4 Uživatelská příručka

V této části se zaměřím na průchody aplikací pro všechny kategorie uživatelů. Většina průchodu pro registrované a standardní uživatele bude totožná, a proto začneme s obecnou funkcionalitou, která je sdílená mezi těmito uživateli.

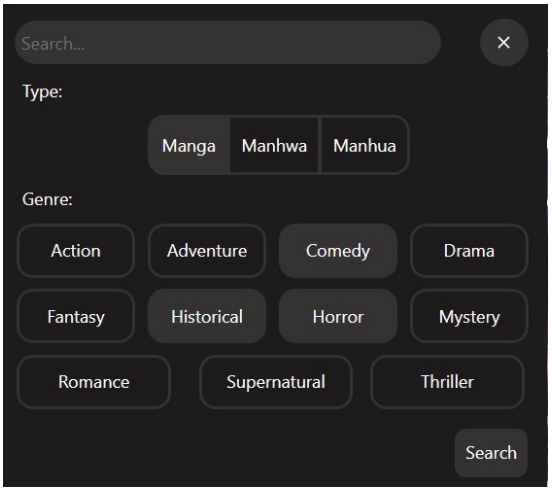
4.1 Navigace

Uživatelé mohou v aplikaci navigovat pomocí nabídky zobrazené na obrázku 3. Tlačítko pro otevření této nabídky se nachází v pravém horním rohu. Po kliknutí na toto tlačítko se nabídka rozvine na celou šířku obrazovky. Registrovaní uživatelé mají navíc přístup k položkám „Profile“ a případně „Editor“.

Obrázek 3: Navigační menu

4.2 Hledání komiksů

Pokud uživatel chce číst konkrétní komiks, může si ho vyhledat pomocí názvu a parametrů v menu, jak je znázorněno na obrázku 4. Menu se zobrazí po kliknutí na ikonu lupy v záhlaví. Uživatel má možnost vybrat jeden typ komiksu a více žánrů. Vybrané žánry a typ jsou zvýrazněny pro lepší orientaci



Obrázek 4: Vyhledávací okno

4.3 Prohlížení komiksů

Na úvodní stránce je zobrazena komponenta „SlideShow“ s populárními komiksy a pod ní je dvacet kartiček nejnovějších titulů. Každá karta obsahuje titulní obrázek, název, informace o třech nejnovějších kapitolách a stav komiksu. Pokud uživatel projeví zájem o prohlížení dalších komiksů, může se přesunout na stránku „comics“ prostřednictvím „menu“, nebo kliknutím na tlačítko „Browse more“. Uživatel má možnost rozhodnout se, zda si přeje rozkliknout konkrétní komiks a získat o něm bližší informace, nebo ihned přejít k čtení. Na stránce s konkrétním komiksem má uživatel možnost seznámit se s základními informacemi o komiksu a jeho komentáři. Dále má možnost v seznamu kapitol kliknout na tlačítko „Start reading“, které ho přesměruje na první kapitolu, nebo si vybrat konkrétní kapitolu. Během čtení jsou uživateli zobrazeny části kapitol postupně za sebou, buď přímo na stránce, nebo v okně (viz obrázek 5), které umožňuje přepínání obrázků podobně jako při čtení knihy. Uživatel se v tomto okně může pohybovat pomocí šipek v záhlaví, kliknutím na pravý nebo levý okraj okna nebo potažením obrázku.

Obrázek 5: Okno pro čtení komiksů ve stylu knihy

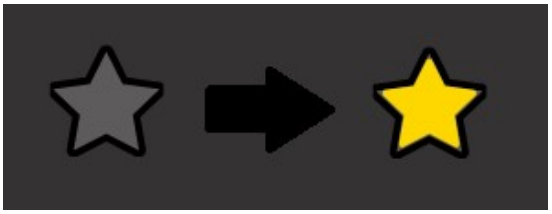
4.4 Registrace a přihlášení

Uživatel se dostane na stránku „přihlášení“ prostřednictvím nabídky v menu, odkud má možnost přejít k registraci nebo obnově hesla. Při registraci je po uživateli vyžadováno, aby zadal platný email, uživatelské jméno, dvakrát zopakoval zvolené heslo s délkou minimálně sedm znaků a uvedl své pohlaví. Pokud registrace proběhne úspěšně, je uživatel přesměrován k „přihlašovacímu formuláři“, kde musí zadat své údaje znovu. V případě zapomenutého hesla má uživatel možnost jej obnovit zadáním svého emailu spojeného s účtem. Na tento email mu poté přijde „resetovacího token“, který si zkopíruje a následně ho s nově zvoleným heslem odesílá pro potvrzení.

4.5 Dodatečná funkcionalita

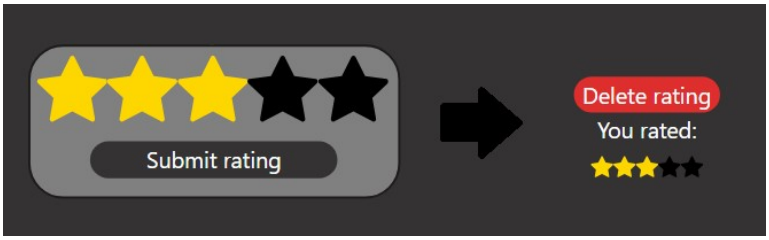
Registrovaní uživatelé mají k dispozici další funkcionality při prohlížení daného komiksu, které zahrnují možnost přidání komiksu mezi své oblíbené, hodnocení a komentování.

- **„Oblíbené:“** Uživatel má možnost kliknout na ikonu hvězdy pro přidání komiksu mezi své oblíbené. Na obrázku 6 je znázorněna animace, během které se ikona hvězdy po kliknutí vyplní zlatou barvou.



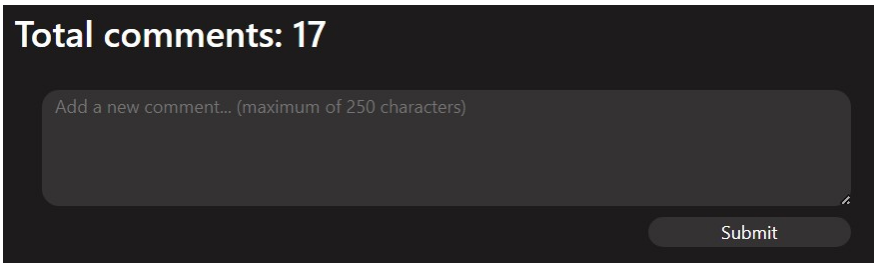
Obrázek 6: Vzhled favorites komponenty

- **„Hodnocení:“** Po kliknutí na tlačítko „Rate me“ se uživateli zobrazí nabídka pěti hvězd. Hvězdy reagují na pohyb kurzoru tím, že změní barvu na zlatou, a zůstanou zlaté v souladu s pořadím vybraných hvězd. Po vybrání hvězdy a odeslání hodnocení se uživateli zobrazí jeho hodnocení a nabídne se mu možnost hodnocení smazat (viz obrázek 7).



Obrázek 7: Vzhled rating komponenty

- **„Komentování:“** Uživatel má možnost přidat komentář k danému komiksu v okně označeném jako „add new comment“ (viz obrázek 8). Délka komentáře je omezena na 250 znaků. Uživatel má také možnost reagovat na již existující komentáře a vyjádřit svůj názor pomocí tlačítek „líbí se“ nebo „nelíbí se“.

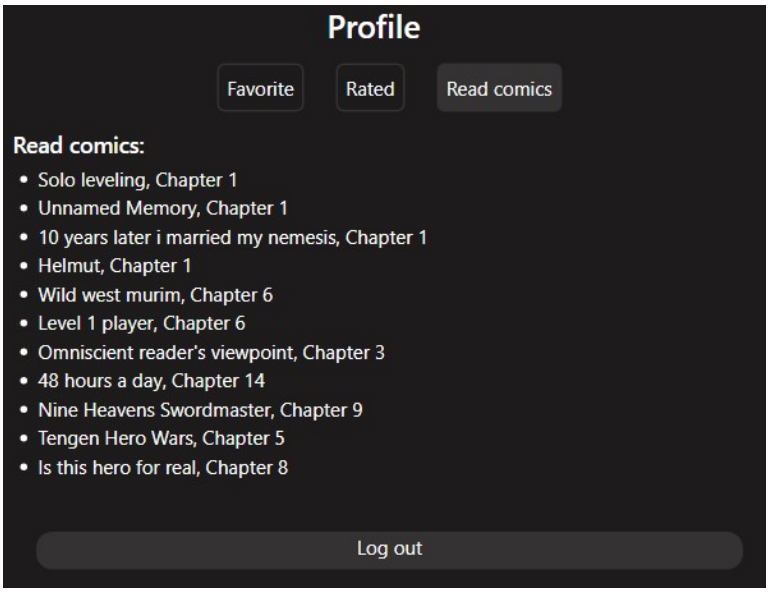


Obrázek 8: Oblast pro komentování

4.6 Profil uživatele

Zde si uživatel může prohlédnout komiksy, s kterými interagoval. Profil je rozdělen do tří částí.

1. Oblíbené komiksy
2. Hodnocené Komiksy
3. Rozečtené komiksy (viz obrázek 9)

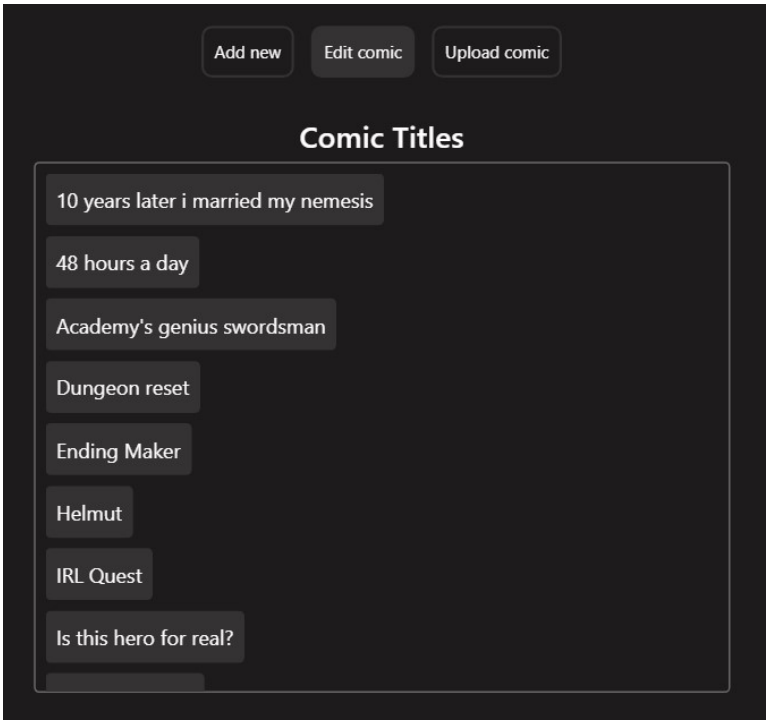


Obrázek 9: Profil uživatele

Ve všech částech má uživatel možnost přejít na konkrétní komiks a u rozečtených komiksů navíc přímo na kapitolu. K dispozici je také tlačítko „Log out“ pro odhlášení uživatele.

4.7 Editor komiksů

Pokud je uživatel editorem, bude mu v menu zpřístupněna položka „Comic editor“. V rámci „editoru“ má uživatel možnost přidávat nové komiksy, upravovat již existující nebo nahrávat nové hromadně. Pokud uživatel zvolí možnost upravit komiks, musí nejprve vybrat požadovaný komiks ze seznamu na obrázku 10.



Obrázek 10: Seznam komiksů v editoru

Dále je uživatel přesměrován na formulář s komiksem. V případě vytváření nového komiksu je tento formulář prázdný, zatímco při úpravě existujícího komiksu jsou v něm již vyplněny informace o vybraném komiksu. Zde má uživatel možnost editovat informace o komiksu a jeho kapitolách. Pro nahrání kapitol slouží zóna pod seznamem. Zde uživatel vloží `.zip` pojmenovaný jako komiks, kterému chce přidat kapitoly a v případě potřeby přidá i titulní obrázek. Soubor by měl obsahovat složky kapitol s jejím číslem, například „chapter1“. V každé složce budou jednotlivé obrázky pojmenovány čtyřmístným číslem s příponou `.jpg`, tedy například „0001.jpg“.

V sekci „Upload comics“ má uživatel možnost stáhnout šablonu komiksu ve formátu `.json`, kterou může následně upravit a nahrát. Pokud uživatel špatně vyplní šablonu, bude upozorněn na přesné místo chyby a bude požádan o opě-

tovné nahrání souboru. Pokud kontrola proběhne úspěšně, zobrazí se uživateli tlačítko „Finish upload“ spolu s počtem připravených komiksů k nahrání.

Závěr

Aplikace umožňuje registrovaným i běžným uživatelům pohodlné čtení komiksů. Při vývoji byl kladen důraz na responzivitu a přehlednost. Registrovaní uživatelé mají k dispozici rozšířenou funkcionalitu, která je zaměřena na zlepšení uživatelského zážitku při čtení. Součástí aplikace je také základní rozhraní navržené pro editory, které umožňuje provádět úpravy a aktualizace záznamů o komiksech, nebo nahrání nových. Po nasazení aplikace existují možnosti pro další rozšíření. Čtenáři mohou například dostávat emaily s oznámením o nově přidaných nebo aktualizovaných komiksech. Další možností je integrace služby Google AdSense, která umožňuje zobrazovat reklamy na webových stránkách a spravovat jejich výdělek.

Conclusions

The application enables both registered and ordinary users to comfortably read comics. Registered users have access to enhanced functionality aimed at improving the reading experience. The application also includes a basic interface designed for editors, allowing them to make edits and updates to comic records, as well as upload new ones. After the application is deployed, there are opportunities for further expansion. For example, readers can receive emails notifying them of newly added or updated comics. Another option is the integration of Google AdSense, which allows displaying advertisements on web pages and managing their revenue.

A. Obsah elektronických dat

`frontend/`
Tento adresář obsahuje všechny soubory a zdrojové kódy frontend serveru.

`backend/`
Tento adresář obsahuje všechny soubory a zdrojové kódy backend serveru.

`redist/`
V adresáři je uložen instalační balíček Node.js.

`install.bat/`
Soubor k doinstalování potřebných balíčků.

`start.bat/`
Soubor k zapnutí obou serverů.

`README.txt/`
Textový soubor s postupem pro instalaci a spuštění aplikace.

`doc/`
Text práce ve formátu PDF, vytvořený s použitím závazného stylu Kidiplom PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový kód textu, vložené obrázky, apod.

Literatura

[1] Mozilla Developer Network. *HTML*. Accessed: April 8, 2024. Dostupný z: ⟨https://developer.mozilla.org/en-US/docs/Web/HTML⟩.

[2] Mozilla Developer Network. *CSS*. Accessed: April 8, 2024. Dostupný z: ⟨https://developer.mozilla.org/en-US/docs/Web/CSS⟩.

[3] Mozilla Developer Network. *CSS basics*. Accessed: April 8, 2024. Dostupný z: ⟨https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics⟩.

[4] Mozilla Developer Network. *JavaScript*. Accessed: April 8, 2024. Dostupný z: ⟨https://developer.mozilla.org/en-US/docs/Web/JavaScript⟩.

[5] Node.js. *Introduction to Node.js*. Accessed: April 8, 2024. Dostupný z: ⟨https://nodejs.org/en/learn/getting-started/introduction-to-nodejs⟩.

[6] Node.js. *An Introduction to the npm Package Manager*. Accessed: April 8, 2024. Dostupný z: ⟨https://nodejs.org/en/learn/getting-started/an-introduction-to-the-npm-package-manager⟩.

[7] Bootstrap Authors. *Bootstrap Documentation*. Accessed: April 8, 2024. Dostupný z: ⟨https://getbootstrap.com/docs/5.3/getting-started/introduction/⟩.

[8] React. *React Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://react.dev/reference/react⟩.

[9] TechTarget. *MongoDB*. Accessed: April 15, 2024. Dostupný z: ⟨https://www.techtarget.com/searchdatamanagement/definition/MongoDB⟩.

[10] Vercel. *Next.js Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://nextjs.org/docs⟩.

[11] Bootstrap, React. *React Bootstrap Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://react-bootstrap.netlify.app/docs/layout/grid⟩.

[12] Icons, React Bootstrap. *React Bootstrap Icons*. Accessed: April 15, 2024. Dostupný z: ⟨https://www.npmjs.com/package/react-bootstrap-icons⟩.

[13] Axios. *Axios Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://axios-http.com/docs/intro⟩.

[14] Express. *Express Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://www.npmjs.com/package/express⟩.

[15] Goode, Troy. *Cors Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://www.npmjs.com/package/cors⟩.

[16] Raut, Nilesh. *How to Configure CORS in Node.js with Express*. Accessed: April 15, 2024. Dostupný z: ⟨https://dev.to/speaklouder/how-to-configure-cors-in-nodejs-with-express-11h⟩.

[17] Karpov, Valeri. *Mongoose Documentation*. Accessed: April 15, 2024. Dostupný z: ⟨https://mongoosejs.com/docs/⟩.

[18] *bcrypt*. Accessed: April 15, 2024. Dostupný z: ⟨https://en.wikipedia.org/wiki/Bcrypt⟩.

[19] *JWT.IO - Introduction to JSON Web Tokens*. Accessed: April 15, 2024. Dostupný z: ⟨https://jwt.io/introduction⟩.

[20] Nodemon. *Nodemon*. Accessed: April 15, 2024. Dostupný z: ⟨https://www.npmjs.com/package/nodemon⟩.

[21] Nodemailer. *Nodemailer*. Accessed: April 27, 2024. Dostupný z: ⟨https://www.npmjs.com/package/nodemailer⟩.

[22] Dotenv. *Dotenv*. Accessed: April 25, 2024. Dostupný z: ⟨https://www.npmjs.com/package/dotenv⟩.

[23] Dropzone, React. *React Dropzone*. Accessed: April 28, 2024. Dostupný z: ⟨https://www.npmjs.com/package/react-dropzone⟩.

[24] adm-zip. *adm-zip*. Accessed: April 28, 2024. Dostupný z: ⟨https://www.npmjs.com/package/adm-zip⟩.

[25] multer. *multer*. Accessed: April 28, 2024. Dostupný z: ⟨https://www.npmjs.com/package/multer⟩.