
F3

Fakulta elektrotechnická
Katedra počítačů

Bakalářská práce

# Evidence skladových zásob skenováním kódů

**Anna Kachmasheva**
Softwarové inženýrství a technologie

Květen 2024
Vedoucí práce: RNDr. Ondřej Žára

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kachmasheva** Jméno: **Anna** Osobní číslo: **503215**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Evidence skladových zásob skenováním kódů**

Název bakalářské práce anglicky:

**Using QR codes for inventory management**

Pokyny pro vypracování:

Nastudujte možnosti skenování čárových/QR kódů pomocí klientských webových API. Prozkoumejte existující knihovny, jejich kompatibilitu a schopnosti.
Navrhněte následně jednoduchou aplikaci, která bude pracovat nad databází položek (skladových zásob) identifikovaných jejich čárovými/QR kódy.
Hlavní vlastnosti aplikace:
- úložtě dat v jednoduché relační databázi (sqlite, mysql, ...)
- REST API pro komunikaci mezi serverem a klientem
- CRUD operace do tabulky skladových zásob (klient získává kódy skenováním)
- plánování výdeje množiny zásob:
- označení podmnožiny existujících zásob
- následný výdej skladových zásob omezený na danou podmnožinu
- kontrola úplnosti výdeje celé podmnožiny
- jednoduchá autorizace (http auth či jméno+heslo)
- uživatelské role:
- anonym: nic
- správce: vše
- uživatel: konfigurovatelná podmnožinu operací
Aplikaci otestuje v různých prohlížečích nad různými druhy kódu. Podrobte ji též kvalitativnímu uživatelskému testování.

Seznam doporučené literatury:

https://github.com/mebjas/html5-qrcode
https://github.com/nimiq/qr-scanner
https://blog.minhazav.dev/research/html5-qrcode

Jméno a pracoviště vedoucí(ho) bakalářské práce:

**RNDr. Ondřej Žára Katedra počítačové grafiky a interakce**

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **07.02.2024** Termín odevzdání bakalářské práce: **24.05.2024**

Platnost zadání bakalářské práce: **21.09.2025**

| RNDr. Ondřej Žára<br>podpis vedoucí(ho) práce | podpis vedoucí(ho) ústavu/katedry | prof. Mgr. Petr Páta, Ph.D.<br>podpis děkana(ky) |
|---|---|---|

## III. PŘEVZETÍ ZADÁNÍ

Studentka bere na vědomí, že je povinna vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____
Datum převzetí zadání

_____
Podpis studentky

# Poděkování / Prohlášení

Děkuji svému vedoucímu tohoto projektu RNDr. Ondřeji Žárovi za podporu při tvorbě této bakalářské práce.

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 24.5.2024

.....

# Abstrakt / Abstract

Práce zkoumá možnosti klientských webových API pro skenování čárových/QR kódů a hodnotí jejich kompatibilitu a možnosti. Součástí je i implementace jednoduché webové aplikace pro správu skladových zásob. Tato aplikace umožňuje skenovat produkty a automaticky aktualizovat databázi zásob, včetně přidávání nových produktů, aktualizace dílů a sledování stavu zásob v reálném čase.

**Klíčová slova:** Skenování čárových kódů, Skenování QR kódů, Webová aplikace, Správa zásob, Vyhledávání produktů pomocí QR kódu

This thesis explores the capabilities of client web APIs for scanning barcodes/QR codes and evaluates their compatibility and capabilities. It also includes the implementation of a simple web application for inventory management. This app allows you to scan products and automatically update your inventory database, including adding new products, updating parts, and tracking inventory in real time.

**Keywords:** Barcode scanning, QR code scanning, Web application, Inventory management, Product search by QR code

# Obsah /

# Tabulky / Obrázky

# Kapitola 1
# Úvod

Účelem projektu je prozkoumat využití čárových/QR kódů pomocí klientských webových API k návrhu implementace systému skladového hospodářství. Informace o skladových zásobách jsou umístěny v relační databázi. Aplikace by měla být implementována tak, aby čárové/QR kódy sloužily k identifikaci jednotlivých výrobků.

Požadavky na finální systém podle zadání jsou: CRUD operace nad tabulkou zboží a objednávek. Součástí aplikace by měla být také kontrola nad skladovými zásobami. Architektura aplikace by měla být typu klient-server s využitím REST API.

Druhá kapitola popisuje existující řešení, která mají webovou aplikaci a využívají čárové/QR kódy k identifikaci zboží a objednávek. Analýza těchto řešení umožní vytvořit požadavky na systém.

Třetí kapitola popisuje analýzu požadavků na systém, která vychází z analýzy stávajících řešení a bakalářského zadání práce. V této kapitole jsou definovány hlavní funkce aplikace a její potenciální uživatelé.

Čtvrtá kapitola se zaměřuje na analýzu současných technologií pro implementaci webové aplikace a byl proveden výběr technologií, které budou použity při implementaci systému.

Pátá kapitola se zaměřuje na prototypy aplikace, jejich vývoj a porovnání s konečným výsledkem. V příloze jsou uvedeny prototypy, které doplňují obsah této kapitoly.

Kapitola 6 popisuje implementaci aplikace, použité technologie a způsob jejich použití v práci. V této kapitole jsou analyzovány knihovny pro snímání kódu s použitim webové kamery.

Sedmá kapitola představuje popis uživatelských testů provedených skupinou osob s cílem zlepšit uživatelský komfort a odstranit kritické chyby při implementaci aplikace.

# Kapitola 2
# Existující řešení

V současné době je k dispozici široká škála aplikací pro řízení skladu. Tyto systémy zahrnují sadu nástrojů a operací, které umožňují společnostem sledovat a řídit skladové operace, jako jsou sledování zásob, příjem, vychystávání a naskladňování. Tyto systémy umožňují podnikům efektivněji využívat pracovní sílu a prostor optimalizací zdrojů a materiálových toků. Většina těchto aplikací je součástí systému ERP.

Pro analýzu existujících systémů pro řízení zásob jsou vybrané systémy, které jsou zaměřené právě na řízení zboží a mají na něj důraz. Výběr hotových řešení je založen na analýze hodnocení aplikací pro správu produktů prezentovaných na webových stránkách[1]. Tyto stránky poskytují informace, které lze využít při výběru softwaru.

Vybrané systémy budou analyzovány podle několika níže uvedených kritérií:

- Funkcionalita aplikace;
- Používání QR/čárových kódů;
- Přehlednost aplikace;
- Vzhled aplikace;
- Nedostatky;
- Hodnocení uživatelů (kontrola na 20.5.24).

## 2.1 Sortly

Jedná se o online aplikace Sortly inventory[2] pro řízení skladů pro malé a střední podniky. Kromě webové aplikace je k dispozici také mobilní aplikace pro zařízení se systémy iOS a Android. Software běží v cloudu. Tato aplikace má bezplatnou verzi, která má oproti placené verzi více omezení. Aplikace se zaměřuje na inventarizaci.

**Funkcionalita aplikace**

Hlavními funkcemi systému jsou řízení pohybu zásob ze skladu do skladu, zobrazení zásob a jejich dodavatelů a sledování polohy zboží. Aplikace umožňuje sledovat nejen umístění zboží, ale také jeho cenu a množství. Kromě toho systém poskytuje funkce pro správu nákupních a dodacích objednávek. Zboží je seskupeno do složek. Ke každé položce lze přidávat poznámky a štítky, které usnadňují vyhledávání a třídění položek. Při vytváření produktu lze přidat vlastní pole a zadat metadata pro tato pole. Uživatelé mohou odesílat upomínky, nastavovat data vrácení zboží nebo vypršení záruky pro pronajaté položky, a dokonce importovat soubory CSV. Aplikace odesílá oznámení, když položky dojdou.

**Používání QR/čárových kódů**

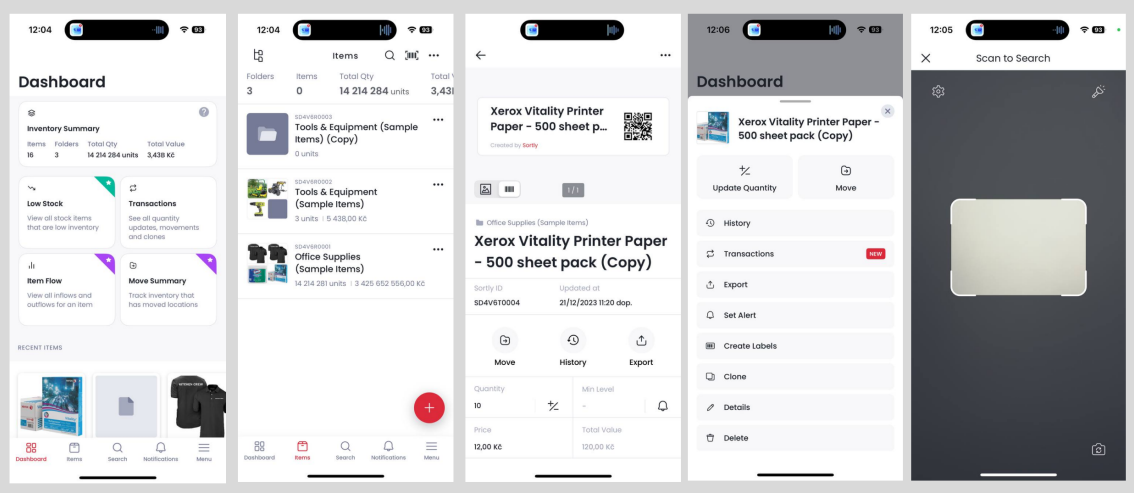
Řešení umožňuje vytvořit QR kód nebo čárový kód pro každý produkt. Vytvořený kód je možné také vytisknout. Při tisku si můžete vybrat velikost kódu z několika nabízených.

[1] https://www.saasworthy.com/
[2] https://www.sortly.com/

V mobilní verzi aplikace je k dispozici vyhledávání zboží pomocí skenování kamerou. Pokud zboží není nalezeno, je zde možnost vytvořit ho ručním zadáním údajů.

**Přehlednost aplikace**

Aplikace je intuitivní a podle recenzí uvedených na webu se snadno používá. Je možné vytvořit vlastní pole pro produkty s vlastními měrnými jednotkami. Příklady stránek z aplikace jsou uvedeny na obrázku níže ??.




**Obrázek 2.1.** Ukázka vzhledu systému Sortly Inventory.

**Nedostatky** Ve webové aplikaci není možnost skenovat položky, tato funkce je k dispozici pouze v mobilní aplikaci. Webová aplikace navíc není v telefonu k dispozici. Pro její používání společnost doporučuje stáhnout si mobilní aplikaci. Není zde možnost sledovat změnu nákladů na zboží v čase. Chybí možnost přidávat pravidelné dodávky.

**Hodnocení uživatelů v systemu na SaasWorthy**: $4.5/5$[3].

**Hodnocení mobilní aplikace v App Store** $4.7/5$[4].

**Hodnocení mobilní aplikace v Google Play** $4.1/5$[5].

## 2.2 Zoho Inventory

Tento systém zahrnuje webové stránky a mobilní aplikaci s podporou systémů iOS a Android. Software funguje v cloudu. Systém umožňuje řídit sklad, prodej, objednávky a zajistit jejich včasné vyřízení.

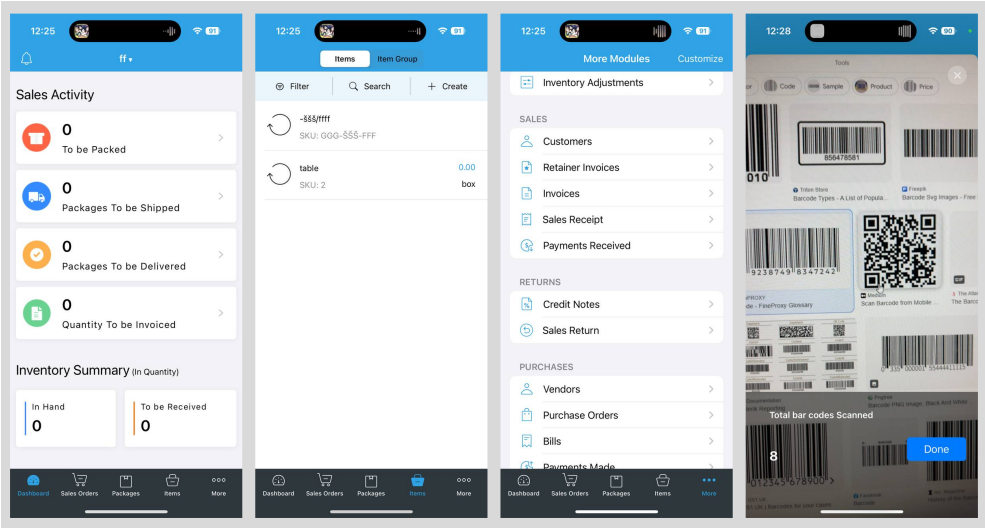
**Funkcionalita aplikace** Systém nabízí vychystávání produktů, vyhledávání, filtrování a sledování sériových zásob a zadávání objednávek. Systém má vlastní cenový model pro produkty. K dispozici je také automatické sledování sériových čísel, šarží, dat spotřeby, objednávek a dodávek. Systém umožňuje export a import výrobků v různých formátech. K dispozici je integrace s dalšími produkty Zoho, jako je CRM a účetnictví, která zajišťuje hladký provoz.

**Používání QR/čárových kódů** Systém je vybaven funkcemi, jako jsou čárové kódy a RFID. Odako tyto funkce podporuje pouze mobilní aplikace. Žádná podpora pro skenování QR kódů.

**Přehlednost aplikace** Uživatelé si obecně pochvalují jednoduchost rozhraní a širokou škálu funkcí, ale abyste mohli plně využívat všechny funkce systému, bude vám trvat

[3] https://www.saasworthy.com/product/sortly
[4] https://apps.apple.com/us/app/sortly-inventory-simplified/id529353551
[5] https://play.google.com/store/search?q=sortly&c=apps&hl=en&gl=US

poměrně dlouho, než se jej naučíte. Příklady stránek z aplikace jsou uvedeny na obrázku níže 2.2.




**Obrázek 2.2.** Ukázka vzhledu systému Zoho Inventory.

**Nedostatky** Podrobná uživatelská příručka není k dispozici. Některé funkce je třeba se naučit. Často se zasekává při stahování velkého množství dat. Špatná podpora, požadavky jsou často odmítány. Používá pouze čárové kódy.

**Hodnocení uživatelů systému na SaasWorthy** $4.5/5$[6].

**Hodnocení mobilní aplikace v App Store** $4.7/5$[7].

**Hodnocení mobilní aplikace v Google Play** $4.8/5$[8].

## 2.3 inFlow Inventory

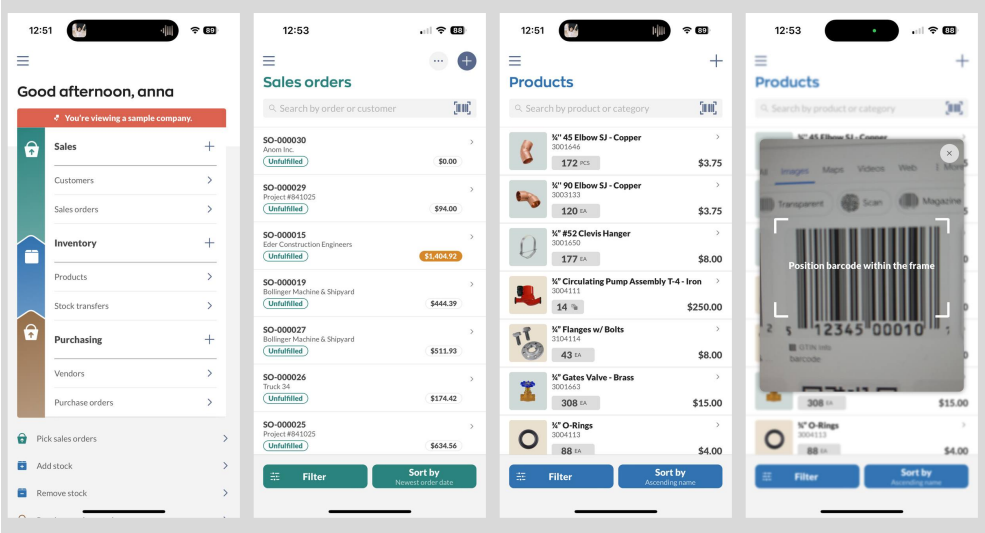
Tento software je k dispozici jako webová aplikace, mobilní aplikace pro iOS a Android a desktopová aplikace pro Windows. Je určen pro malé a střední firmy na správu zásob.

**Funkcionalita aplikace** Umožňuje spravovat zboží, objednávky a nákupy, včetně správy informací o zákaznících a dodavatelích. Systém poskytuje nástroje pro prognózování zásob, sledování nákladů a vyhledávání položek a objednávek podle různých filtrů. Obsahuje vlastní cenové modely. Disponuje nástroji pro různé přehledy.

**Používání QR/čárových kódů** Čtení čárových kódů v mobilní aplikaci s podporou RFID.

**Přehlednost aplikace** Jednoduchý design aplikace a dostatečně široká funkčnost. Ve srovnání s konkurenčními aplikacemi je však design zastaralý. Příklady stránek z aplikace jsou uvedeny na obrázku níže 2.3.

[6] https://www.saasworthy.com/product/zoho-inventory
[7] https://play.google.com/store/search?q=zoho+inventory&c=apps&hl=en&gl=US
[8] https://apps.apple.com/us/app/zoho-inventory-management-app/id1037960494




**Obrázek 2.3.** Ukázka vzhledu systému inFlow.

**Nedostatky** Nepodporuje QR kódy. Čtení čárových kódů je k dispozici pouze v mobilní aplikaci. Čárový kód musí být při skenování přesně umístěn v rámečku, což od uživatele vyžaduje další úsilí. Desktopová aplikace nepodporuje operační systém Linux. Nedostatečný přístup k rozhraní API.

**Hodnocení uživatelů systému na SaasWorthy** $4.5/5$[9].

**Hodnocení mobilní aplikace v App Store** $4.2/5$[10].

**Hodnocení mobilní aplikace v Google Play** $3.6/5$[11].

## 2.4 Závěr

Po analýze stávajících řešení lze konstatovat, že hlavní problémy jsou:

- Absence možnosti skenovat QR kódy nebo čárové kódy pomocí webové kamery ve všech třech prezentovaných systémech. Inventární systém Sortly inventory podporuje ve srovnání s ostatními dvěma systémy nejen čárové kódy, ale i QR kódy.
- Systémy Zoho inventory a inFlow jsou poměrně složité na používání, přestože jsou určeny pro malé a střední podniky, a je nutné si přečíst příručku, aby bylo možné plně využít všechny jejich potřebné funkce.
- Neintuitivní design.

Na základě této analýzy byla jako řešení pro práci zvolena aplikace s možností skenování kódu na mobilních zařízeních. Webové aplikace pro skenování QR kódu, které běží přímo v prohlížeči, poskytují jedinečné výhody oproti jiným metodám skenování, jako jsou mobilní aplikace nebo vyhrazené skenery. Jsou dostupné na jakémkoli zařízení s webovým prohlížečem, což z nich činí univerzální řešení pro uživatele s různými typy zařízení, včetně telefonů, tabletů a stolních počítačů. Kromě toho používání webových aplikací snižuje potřebu fyzických skenerů a dalšího vybavení, což pomáhá snížit elektronický odpad a celkové náklady.

Zvláštní pozornost bude věnována také jednoduchosti a intuitivnosti designu.

[9] `https://www.saasworthy.com/product/inflow-inventory`
[10] `https://apps.apple.com/us/app/inflow-cloud-companion-app/id1397589892`
[11] `https://play.google.com/store/search?q=inflow+inventory&c=apps&hl=en&gl=US`

# Kapitola 3
# Analýza požadavků

Aplikace vyvinutá v rámci práce musí uživatelům poskytovat určité funkce. Na základě zadání byly stanoveny funkční a nefunkční požadavky na systém.

## 3.1 Analýza potenciálních uživatelů

Aplikace je primárně určena pro skladníky. Funkčnost aplikace je určena jak pro pracovníky skladu, kteří zodpovídají za příjem a výdej zboží, tak pro administrátory, kteří vytvářejí objednávky.

## 3.2 Funkční požadavky

### 3.2.1 Autorizace a registrace

- FR01: Systém umožní uživateli registraci. Požadované informace jsou e-mail a heslo. E-mail musí být pro každého uživatele jedinečný. V případě potřeby může uživatel uvést své jméno a příjmení.
- FR02: Systém umožní uživateli přihlásit se do aplikace pomocí e-mailu a hesla.

### 3.2.2 Správa osobních údajů

- FR03: Systém umožní uživateli nahlížet do jeho osobních údajů.
- FR04: Aplikace umožní uživateli změnit jeho osobní údaje, jako je e-mail, heslo, jméno, příjmení a jeho role.
- FR05: Uživatel může smazat svůj osobní profil.
- FR06: Systém umožní správu účtů jiných uživatelů, včetně jejich zakládání a mazání.

### 3.2.3 Řízení skladových zásob

- FR07: Systém umožní prohlížet všechny produkty. Informace o produktu zahrnují kromě jeho názvu také kategorii, popis a stav produktu.
- FR08: Systém umožní najít produkt naskenováním kódu.
- FR09: Aplikace umožní uživateli upravit produkt nebo jej v případě potřeby smazat.
- FR10: Systém umožní vytvořit objednávku. Objednávka může obsahovat více produktů.
- FR11: Aplikace umožní spravovat objednávku a měnit její stav. Během zpracování objednávky systém umožní použít kameru ke skenování kódů produktů.
- FR12: Systém umožní vytisknout QR kód produktu.

## 3.3 Nefunkční požadavky

- NFR01: Systém bude implementován jako webová aplikace.
- NFR02: K ukládání dat bude použita relační databáze.
- NFR03: Pro komunikaci mezi serverem a klientem bude použito REST API. Data jsou odesílána ve formátu JSON.
- NFR04: Pro použití skenování bude systém vyžadovat přístup k webové kameře.
- NFR05: Systém bude kompatibilní s prohlížeči Google Chrome a Safari.

## 3.4 Případy užití (Use Case)

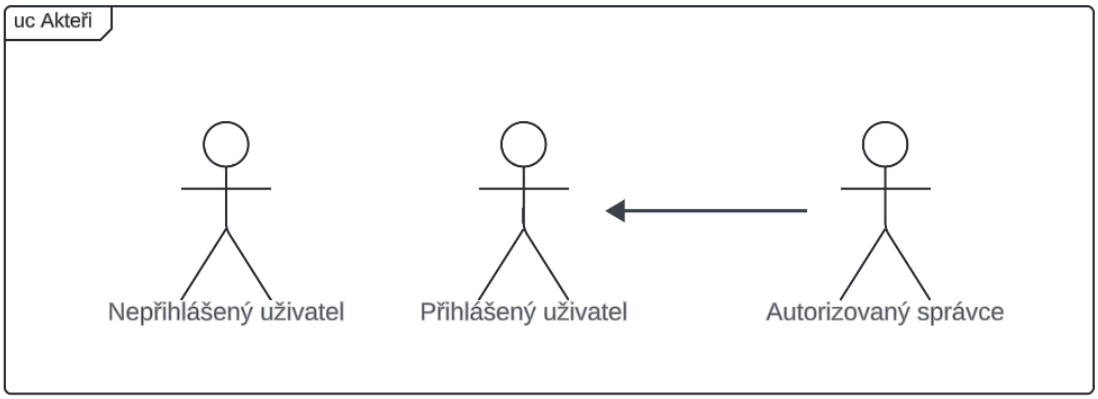
Na základě výše uvedených požadavků na systém byly vytvořeny případy užití.

### 3.4.1 Aktéři

**Nepřihlášený uživatel** - má možnost se do systému zaregistrovat, nebo přihlásit.

**Přihlášený uživatel** - má omezení při používání systému. Systém mu nabízí funkce, jako jsou správa osobních údajů, vyhledávání produktů v databázi naskenováním kódu, prohlížení dostupných objednávek a zpracování objednávek.

**Autorizovaný správce** - má neomezené možnosti. Oproti běžnému uživateli umí vytvářet a spravovat produkty, objednávky a uživatele systému.




**Obrázek 3.1.** Diagram UC. Aktéři.

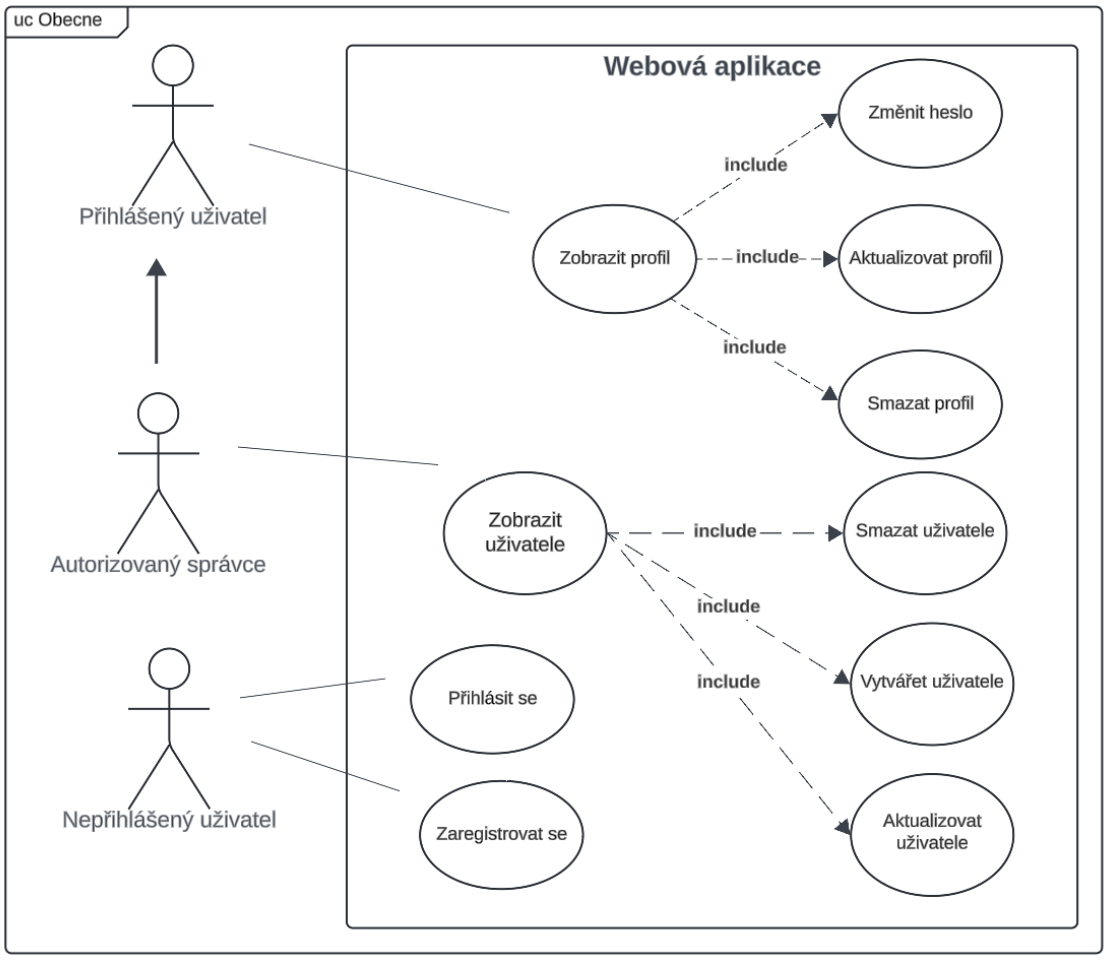
## 3.5 Obecné UC

Systém umožňuje registraci neoprávněného uživatele (FR01). E-mail uživatele musí být v rámci systému jedinečný. Heslo musí být silné. Pokud uživatel při registraci zadá neplatné údaje, systém zobrazí chyby a neumožní mu registraci, dokud nebudou všechny chyby opraveny.

Po úspěšné registraci bude uživatel automaticky autorizován v systému. Při přihlašování do systému musí neoprávněný uživatel zadat e-mail a heslo (FR02). Pokud uživatel s takovými údaji není v databázi nalezen, systém zobrazí chybu. Na přání si uživatel při přihlašování do systému může uložit přihlašovací údaje pro budoucnost do prohlížeče.

Systém umožní oprávněnému uživateli nahlížet do jeho osobních údajů (FR03), měnit je (FR04) nebo smazat jeho účet (FR05).

Pro oprávněného uživatele s administrátorskými právy systém umožní vytvářet uživatele a spravovat jejich osobní účty (FR06).



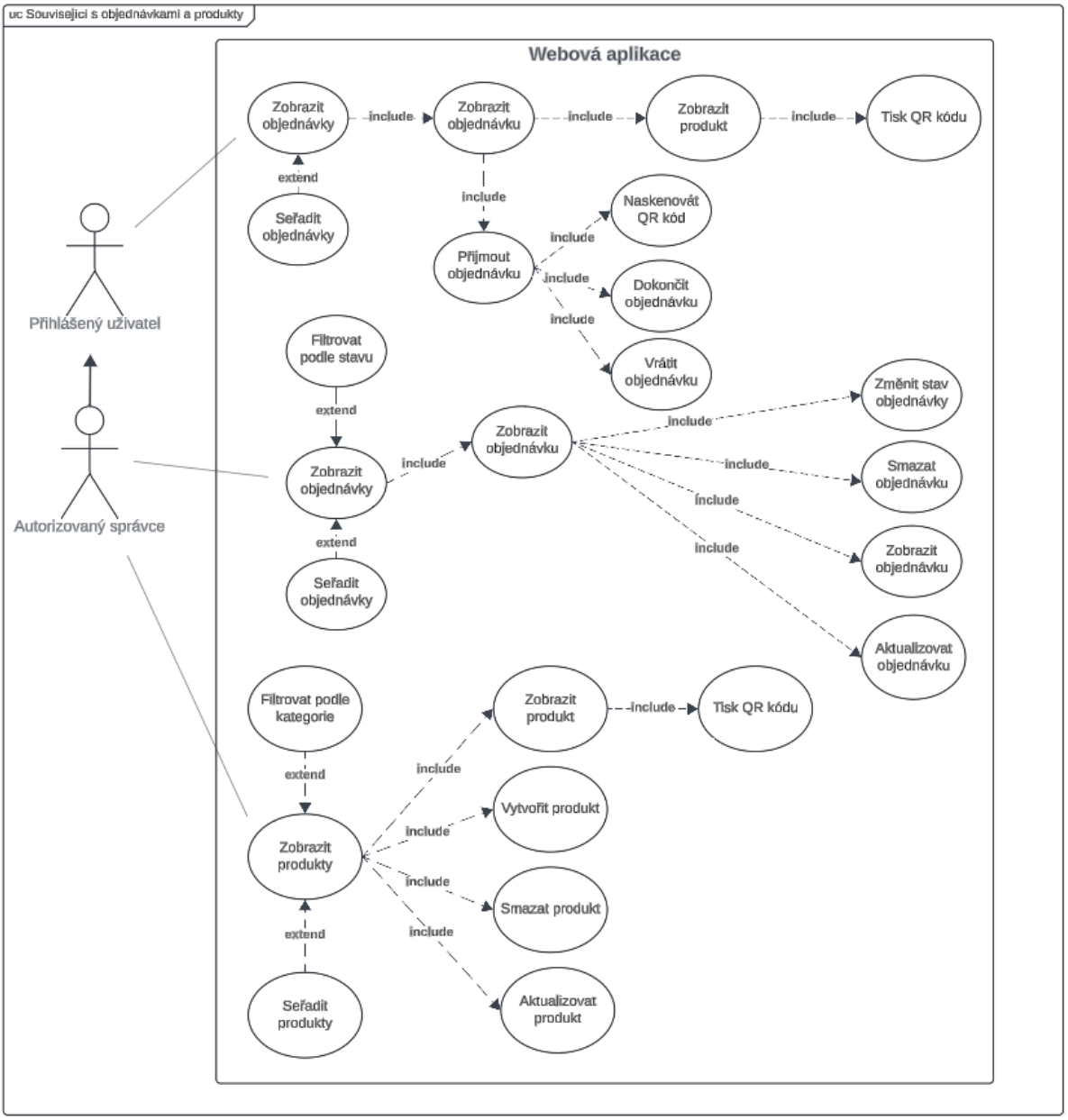

**Obrázek 3.2.** UC související s objednávkami a produkty.

## 3.6 UC související s objednávkami a produkty

Systém poskytuje správci možnost vytvářet, aktualizovat, mazat produkty a zobrazovat tabulku se všemi produkty (FR09). Administrátor je také může zobrazovat podle kategorií a třídit výsledný seznam podle názvu, data jeho poslední aktualizace (FR07). Uživatel si může vytisknout QR kód pro každý produkt (FR12).

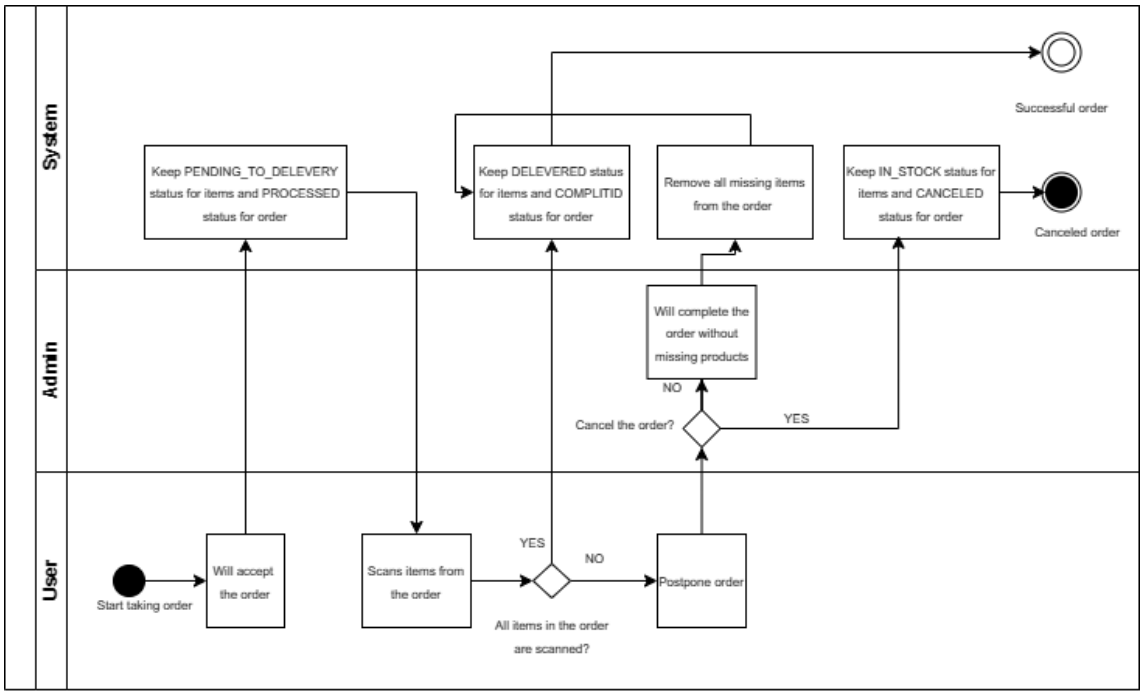
Systém umožňuje správci vytvářet objednávky, které obsahují seznam produktů (FR10). Oprávnění uživatelé si ho mohou prohlížet, filtrovat podle stavu a třídit.

Objednávky vytvořené administrátorem mohou zpracovávat běžní oprávnění uživatelé. Při zpracování objednávky systém změní stav produktů z objednávky. Při sestavování zakázky umožňuje uživateli použít kameru ke skenování kódů (FR11). Pokud byly všechny produkty z objednávky naskenovány uživatelem, může běžný uživatel převést objednávku do dalšího stavu a vydat produkty ze skladu. Pokud ale některé nebyly na skladě nalezeny nebo nebyly naskenovány, uživatel může při jejim sestavování převést objednávku do stavu odložené objednávky. Další stav nevyřízených objednávek může administrátor změnit (FR11), a to bud' dokončit objednávku a odstranit z ní všechny chybějící produkty, nebo objednávku zrušit.

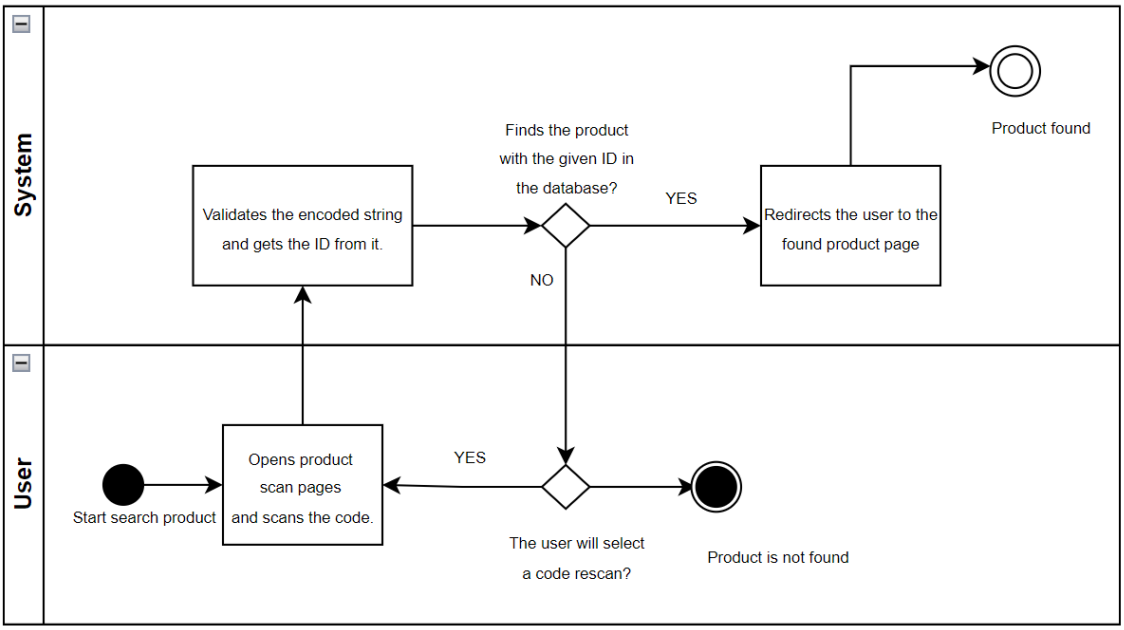



**Obrázek 3.3.** Diagram UC související s objednávkami a produkty.

## 3.7 Hlavní process




**Obrázek 3.4.** Proces zpracování objednávky.

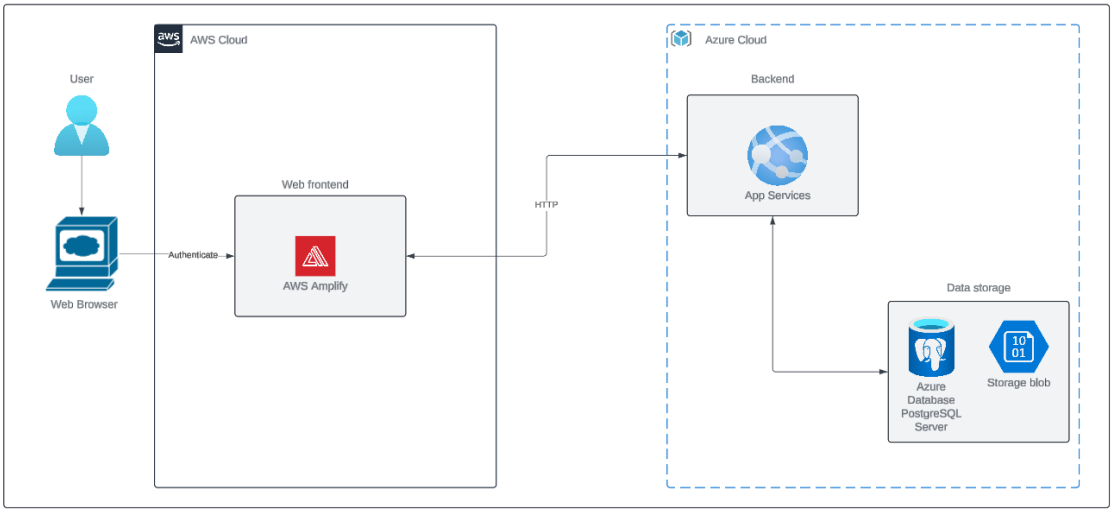



**Obrázek 3.5.** Proces vyhledávání produktů.

# Kapitola 4
# Návrh řešení

## 4.1 Architektura webové aplikace

Vzhledem k tomu, že má systém poměrně jednoduchou strukturu, aby se zabránilo přílišnému komplikování jeho návrhu pro implementaci aplikace, byla zvolena monolitická architektura [1]. Backend aplikace je napsána v jazyce Java [2] pomocí frameworku Spring Boot, frontend je napsán v jazyce JavaScript pomocí knihovny ReactJS. Obě komponenty systému byly podle schématu nasazeny v cloudovém úložišti a komunikují spolu prostřednictvím požadavků HTTP 4.1.




**Obrázek 4.1.** Architektura aplikace.

## 4.2 Backend

### 4.2.1 Java

Pro vývoj backendu byl zvolen jazyk Java a framework Spring Boot [3]. Java je jedním z nejoblíbenějších programovacích jazyků na světě s obrovskou komunitou a technickou podporou, což zaručuje stabilitu práce po dlouhou dobu. Volba frameworku Spring Boot vychází ze skutečnosti, že tento framework umožňuje automatizovat procesy konfigurace a sestavení podnikové aplikace, čímž se urychluje a snižují náklady na vývoj aplikace.

### 4.2.2 Postgresql

Jako úložiště informací byla zvolena relační databáze PostgreSQL [4]. Poskytuje možnost ukládat data strukturovaným způsobem a zajišťuje bezpečnost operací díky vlastnostem ACID [5]. V současné době je PostgreSQL nejoblíbenější relační open-source databází.

### 4.2.3 Azure cloud

Pro hostování backendové součásti projektu byla vybrána služba Azure. Azure je cloudová platforma společnosti Microsoft, která nabízí různé služby IaaS a PaaS pro vývojáře a podniky. Volba Azure je dána především jednoduchostí nasazení aplikací Spring pomocí služby Azure Spring Apps [6]. V cloudu Azure jsou hostovány i takové části aplikace, jako je databáze a úložiště obrazů, což umožňuje bezpečný pohyb dat v rámci jednoho systému.

## 4.3 Frontend

### 4.3.1 JavaScript a React

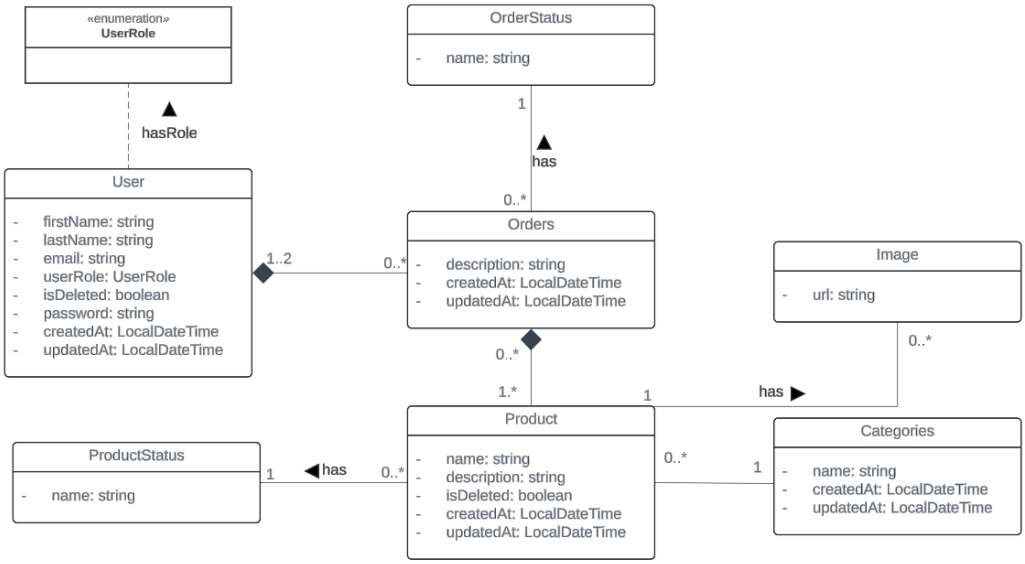
JavaScript je jazyk, který funguje ve všech moderních prohlížečích a je standardem v současné době pro vývoj webových stránek. Většina webových technologií je optimalizována pro práci s ním. Proto volba jazyka pro implementaci frontendu pipadla na tento jazyk. Pro zjednodušení vývoje aplikace byla zvolena knihovna ReactJS [7], která umožňuje používat komponentový přístup a znovu používat existující komponenty v různých částech aplikace.

### 4.3.2 AWS Amplify

Jako platforma pro frontendový hosting byla zvolena služba AWS - AWS Amplify [8]. Tato služba je speciálně navržena pro urychlení vývoje webových aplikací, což ji odlišuje od cloudových úložišť, jako je Azure, pro frontend hosting. Hlavní výhodou tohoto systému jsou procesy CI/CD s plným cyklem, možnost připojit větev systému GIT a automaticky nasadit veškeré změny v uživatelském rozhraní.

## 4.4 Diagram tříd

Společně s UC diagramy byl vytvořen tento diagram, který je uveden na obrázku 4.2.




**Obrázek 4.2.** Doménový model.

Hlavními objekty modelu 4.2 jsou User, Product a Order. Každý z těchto tří objektů má datum a čas vytvoření a datum a čas poslední aktualizace. User může mít jednu z rolí. Je to buď správce, nebo běžný uživatel. Role určuje, které funkce aplikace bude moci uživatel používat. Uživatel s přístupovými právy správce může vytvářet Product a Order a je jejich tvůrcem. Uživatel s běžnými přístupovými právy může pracovat s objednávkami a je jejich zpracovatelem. Order má tedy alespoň jednoho uživatele, který je jejím tvůrcem, a může mít uživatele, který je jejím zpracovatelem. Objednávka obsahuje minimálně 1 produkt. Objednávka i produkt mají svůj vlastní stav, který je definován názvem, jenž je pro tuto kategorii stavů jedinečný. Uživatel a produkt mají informace o tom, zda je objekt smazán. To se provádí kvůli zachování konzistence dat.

# Kapitola 5
# Prototyp

Vývoj prototypu byl zahájen Lo-fi prototypy B, které byly dále rozpracovány do prototypů Hi-fi C a nakonec byl realizován frontend webové aplikace. Prototypy byly vytvořeny pomocí programu Figma[1].

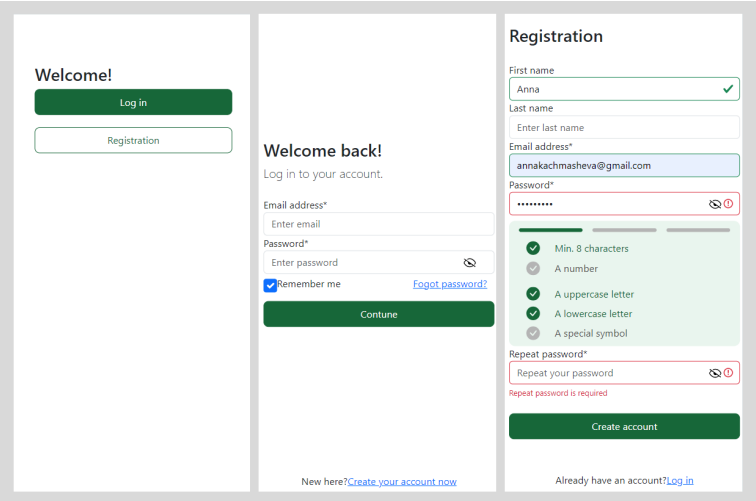
Aplikace je rozdělena na 5 hlavních stránek, které jsou přístupné z postranního panelu, a z některých je přechod na další stránku. Například pokud uživatel klikne na produkt ze seznamem produktů, bude přesměrován na stránku produktu. Podobně pokud klikne na objednávku ze stránky s objednávkou, systém uživateli zobrazí objednávku, a pokud klikne na produkt z této objednávky, zobrazí se konkrétní produkt.

Aplikace používá modální okna pro vytvoření nebo aktualizaci nového produktu, objednávky nebo uživatele. Modální okna se používají také k potvrzení vymazání a ke skenování QR kódů, když uživatel zpracovává objednávku.

Byl zvolen minimalistický design, barevné schéma a písmo. Aby se minimalizovaly chyby a usnadnila se orientace v aplikaci, byly přidány další nápovědy a tipy.

## 5.1 Domovská stránka, přihlášení a registrace

Neautorizovaný uživatel má přístup pouze k domovské stránce, přihlašovacím a registračním stránkám. Každý z formulářů má svá vlastní validační pravidla, pokud nejsou splněna, pak je pod polem uvedena chyba a nápověda pro správné vyplnění pole. Dokud nejsou údaje platné, systém uživateli neumožní vstup do aplikace. Přihlašovací stránka má možnost zapamatovat si přihlašovací údaje.



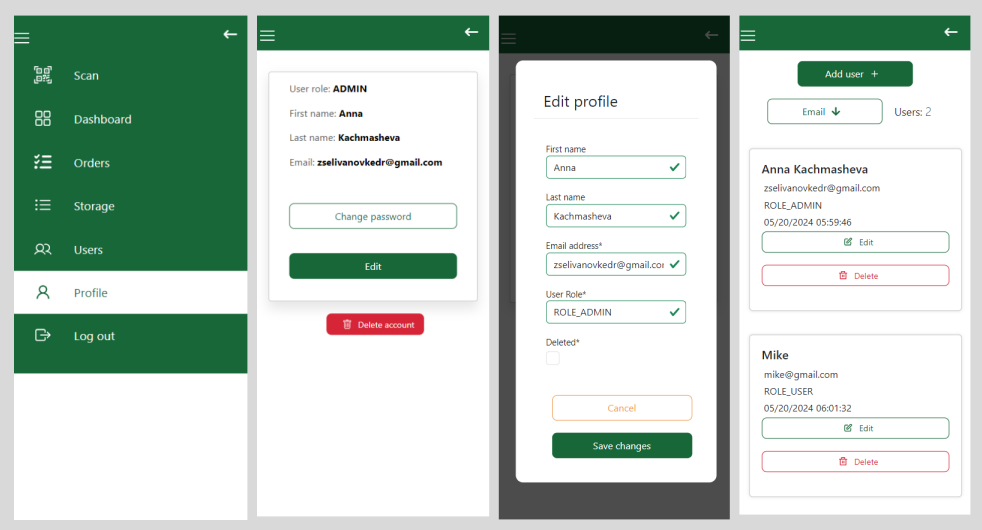

**Obrázek 5.1.** Stránky pro registraci a přihlášení.

[1] https://www.figma.com/design/FuUuUEmxrRD8xFPQ4piWXh/bp?node-id=2

## 5.2 Stránky profilů a seznamů uživatelů

Každý oprávněný uživatel má možnost zobrazit své osobní údaje uvedené při registraci a také je změnit nebo v případě potřeby profil vymazat. K dispozici jsou modální okna pro aktualizaci a mazání profilů.

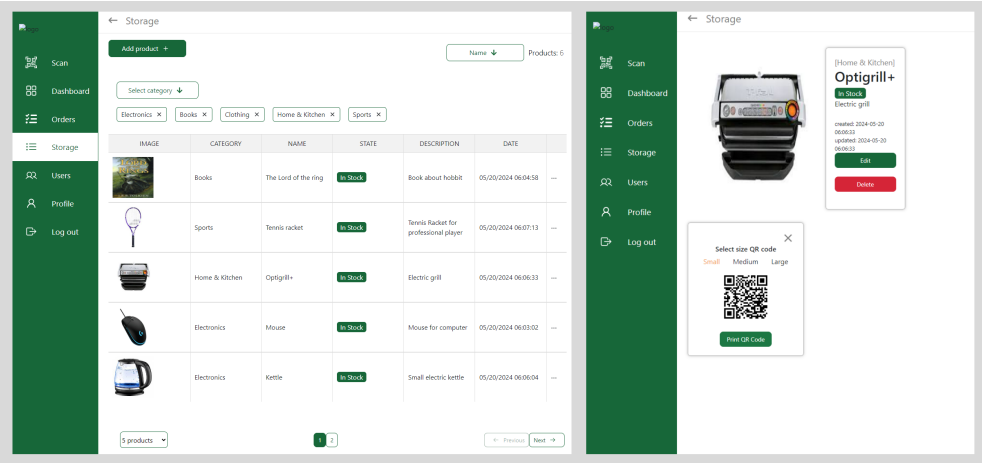
Správce může zobrazit seznam všech uživatelů a spravovat je. Stránka se seznamem uživatelů má stránkování.




**Obrázek 5.2.** Menu, stránka uživatele a stránka profilu.

## 5.3 Stránka se seznamem produktů

Stránku produktu může zobrazit oprávněný uživatel s právy správce. Každý produkt má barevný štítek, který zobrazuje aktuální stav produktu. Uživatel může produkty třídit a filtrovat. Tato stránka má stránkování. K dispozici jsou také modální okna pro přidávání, úpravy a mazání produktů. Je možné kliknout na konkrétní výrobek a přejít na stránku s jeho popisem.



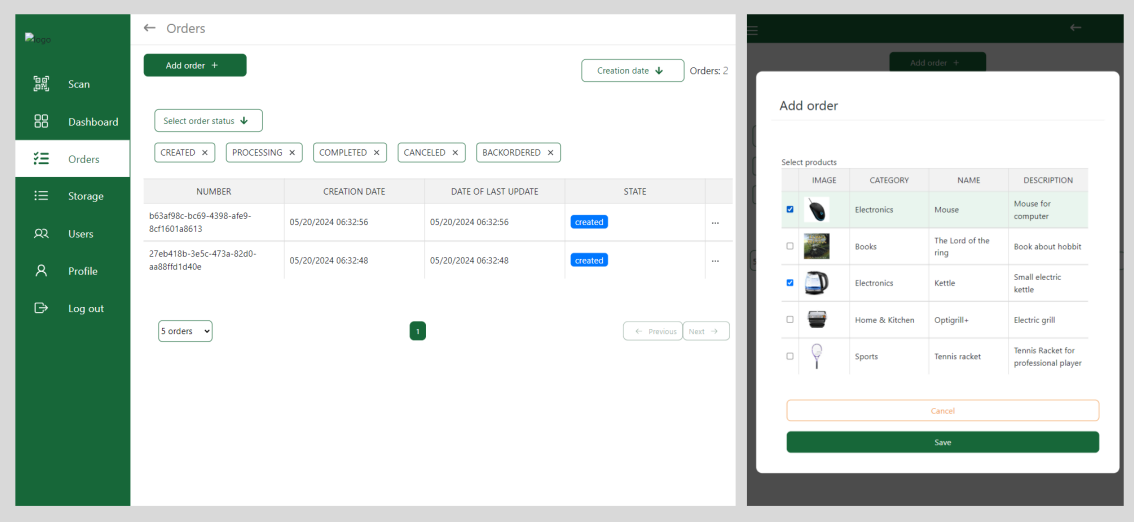

**Obrázek 5.3.** Seznam produktů a stránka produktu.

Na stránce konkrétního produktu je kromě informací o produktu k dispozici také tisk QR kódu s výběrem velikosti kódu.

## 5.4 Stránka se seznamem objednávek

Stránka objednávky je jako stránka se seznamem produktů. Stejně jako stránka produktu má filtry, třídění a stránkování. Aktuální stavy objednávek jsou také přehledně označeny různými barvami. Ze stránky se seznamem objednávek je možné přejít na stránku s konkrétní objednávkou a poté na stránku s produktem z objednávky.

Na stránce objednávky systém poskytuje uživateli a správci různá tlačítka pro správu stavu produktu v závislosti na stavu objednávky a dostupnosti produktů v objednávce.




**Obrázek 5.4.** Seznam objednávek a formulář pro vytvoření objednávky

# Kapitola 6
# Implementace

## 6.1 Implementace aplikace

Backendová část byla implementována na platformě Spring Boot. Po otestování funkčnosti na lokálním serveru byla aplikace nasazena na cloud Azure pomocí služby Azure Spring Apps. Pro správu databáze byla použita jedna z implementací JPA Hibernate, která umožňuje vyhnout se použití dalších systémů pro správu databází, jako je Liquibase, a vytvářet tabulky na základě javovských tříd. Na serveru Azure byla také umístěna databáze PostgreSQL a úložiště blob.

Frontendová část byla implementována v jazyce JavaScript pomocí knihovny React. Na základě testů s knihovnami pro QR kódy bylo rozhodnuto zvolit knihovnu react-qr--reader. Po implementaci a otestování aplikace na několika typech zařízení a prohlížečů byl frontend umístěn na cloud AWS pomocí služby AWS Amplify.

### 6.1.1 Datová vrstva(DAO)

V aplikaci Spring Boot se DAO používá k interakci s databází. Spring boot k tomuto účelu používá JPA. Nejběžnější implementací JPA je Hibernate [9]. Hibernate podporuje mnoho moderních databází, což umožňuje jejich nahrazení s minimálními změnami kódu. Hibernate automaticky generuje dotazy SQL na základě operací s objekty Java, což minimalizuje chyby při psaní složitých dotazů SQL. Jednou z výhod Hibernate je podpora ukládání do mezipaměti, která pomáhá snížit zatížení databáze a urychlit provádění dotazů.

### 6.1.2 Aplikační vrstva

Veškerá aplikační logika se nachází v aplikační vrstvě, kde se manipuluje s přijatými daty, zpracovávají se a propojují s daty z databáze. Po úspěšném zpracování dat se data vrátí do prezentační vrstvy, která je vrátí klientovi.

### 6.1.3 Prezentační vrstva

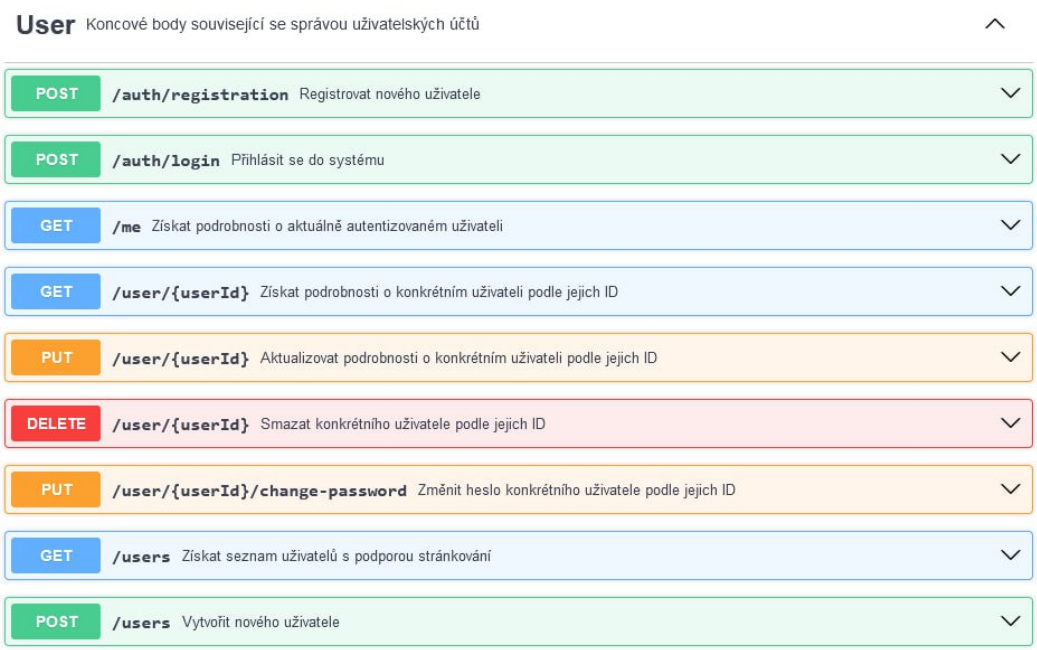
Prezentační vrstva je část architektury aplikace, která je zodpovědná za interakci s uživatelem. V případě aplikace Spring Boot použité k implementaci funkčních požadavků je tato vrstva metodou, která přijímá data ve formátu JSON [10] na adresách URL. Jakmile jsou data přijata, jsou předána do obchodní vrstvy, kde jsou zpracována na základě požadované funkčnosti požadavku. Po zpracování jsou v závislosti na požadavku buď uložena do databáze, nebo vrácena uživateli ve formátu JSON.

### 6.1.4 API pro komunikaci se serverem

Aplikace obsahuje implementaci rozhraní API [1] pro interakci se serverem. S ohledem na popsané funkční požadavky byly koncové body rozděleny do čtyř hlavních skupin: User, Product, Category, Order.

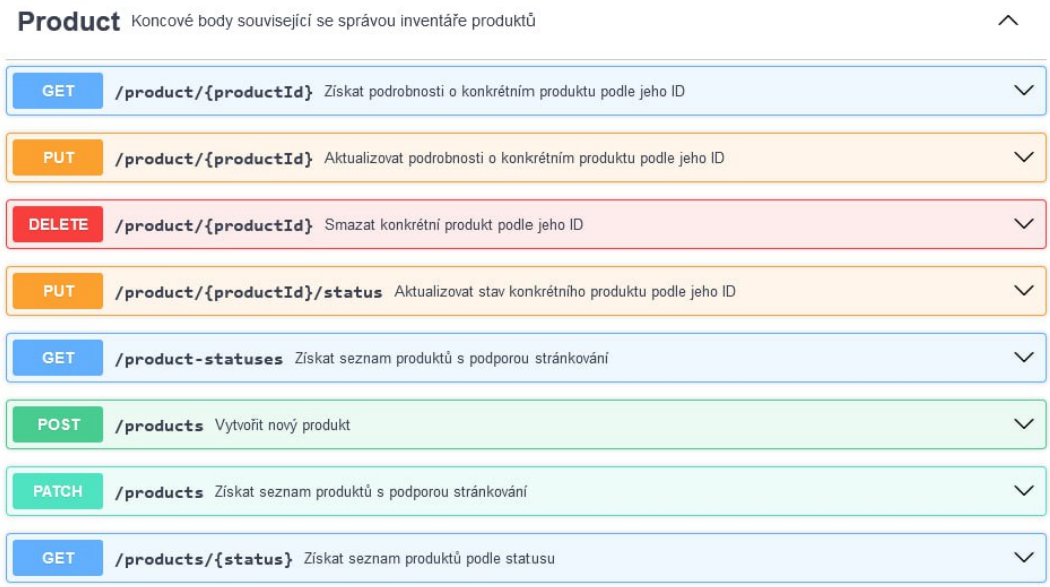
[1] `https://app.swaggerhub.com/apis/ANNAKACHMASHEVA_1/inventory_track/1.0.0`

1. User – Koncové body související se správou uživatelských účtů 6.1.



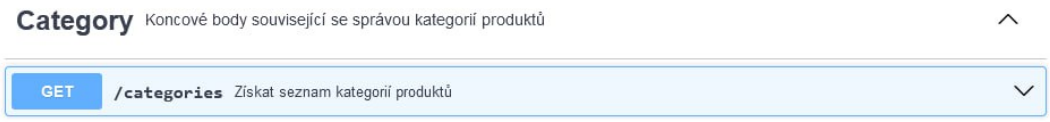

**Obrázek 6.1.** Rozhraní pro správu uživatele.

2. Product – Koncové body související se správou zásob produktů 6.2.



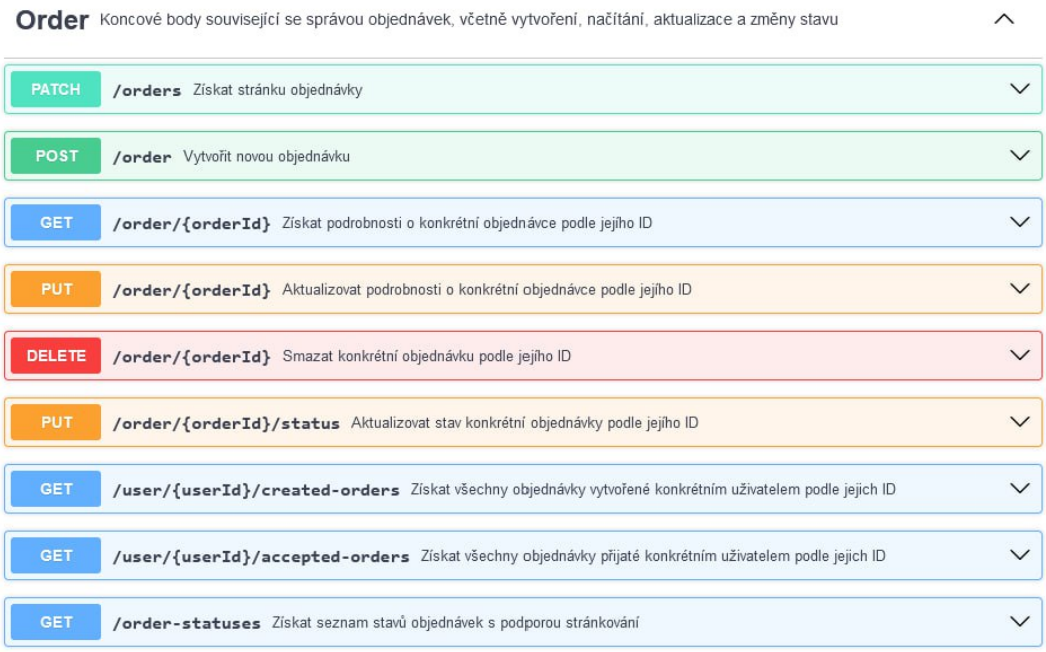

**Obrázek 6.2.** Rozhraní pro správu zásob.

3. Category – Koncové body související se správou kategorií produktů 6.3.




**Obrázek 6.3.** Rozhraní pro správu kategorií produktů.

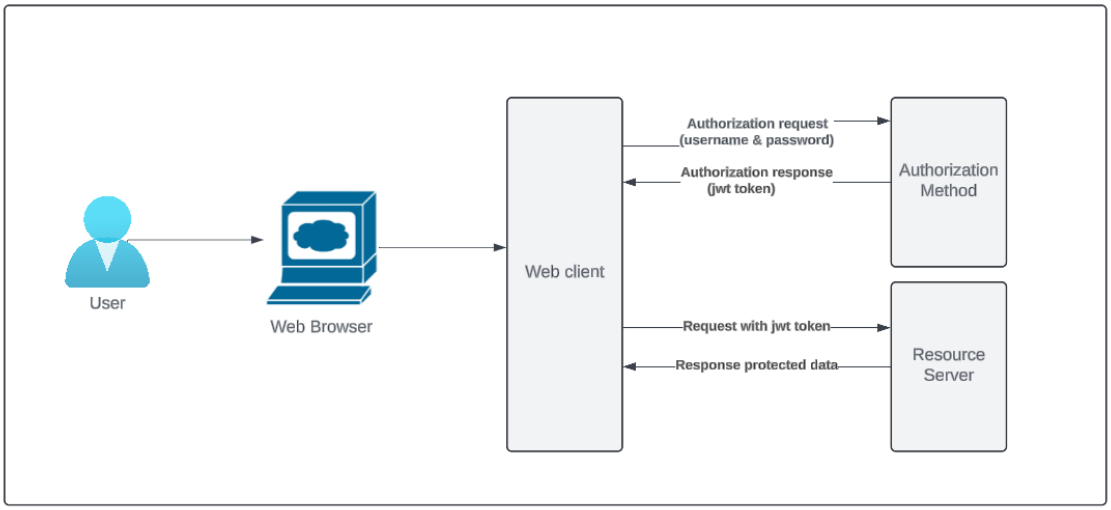
4. Order – Koncové body související se správou objednávek, včetně vytváření, vyhledávání, aktualizace a změny stavu 6.4.




**Obrázek 6.4.** Rozhraní pro správu objednávek.

### 6.1.5 Autentizace a autorizace

Aplikace používá token JWT [11], který chrání koncové body rozhraní REST API 6.5. Uživatel obdrží token po autorizaci, zadáním platného přihlašovacího jména a hesla. Server při pokusu o volání chráněných koncových bodů bez tokenu vrátí chybu `403 Forbidden`, což je standardní kód odpovědi HTTP pro odepření přístupu k požadovanému prostředku. Po autorizaci je token JWT uložen do úložiště localStorage na straně frontendu a předán do hlaviček v následných dotazech.

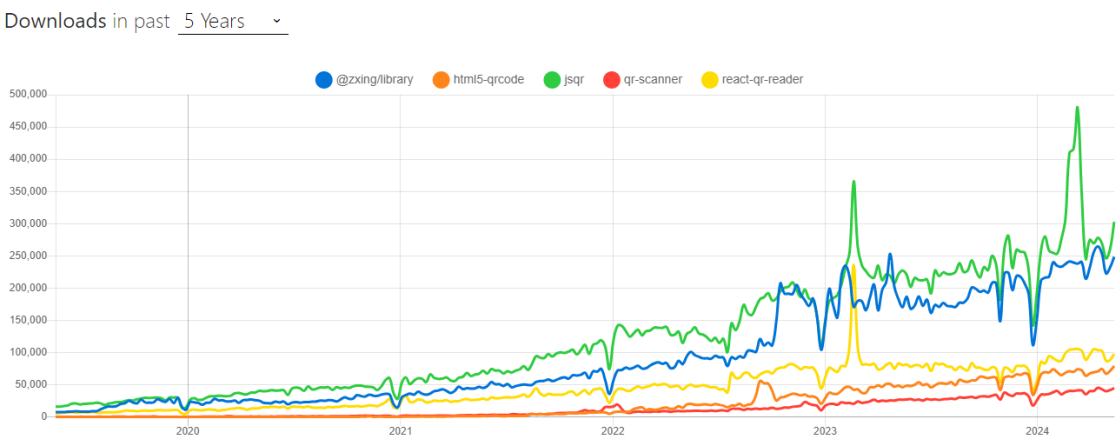



**Obrázek 6.5.** Autorizační proces

## 6.2 JS knihovny pro skenování QR kódů

Tato kapitola se zaměřuje na představení a analýzu několika JS knihoven určených pro skenování QR kódů. Budou prozkoumány jejich hlavní charakteristiky, funkčnost, výhody a možná omezení. Dále budou uvedeny příklady implementace těchto knihoven ve webové aplikaci pro správu zboží ve skladu. Tyto příklady budou testovány a vzájemně porovnávány.

Byly analyzovány trendy ve využívání knihoven [2]. Jsou znázorněny na obrázku 6.6. Na základě těchto statistik bylo vybráno 5 nejčastěji používaných knihoven.




**Obrázek 6.6.** Populární JS knihovny pro skenování QR kódů

### 6.2.1 jsqr

Knihovna jsQR [12] je čistě JS open source knihovna pro čtení QR kódu navržená tak, aby byla flexibilní ve více prostředích. Dokáže zpracovat surové obrázky a je schopna najít, extrahovat a analyzovat jakékoli QR kódy, které se v nich nacházejí. Díky tomu je ideální volbou pro projekty, kde potřebujete skenovat QR kódy z různých zdrojů, jako je stream z webové kamery, obrázek nahraný uživatelem, nebo dokonce uvnitř interního procesu Node.js. Knihovna je zcela samostatná a nezávisí na kódu specifickém pro platformu.

Při dekódování QR kódu může vracet různá data, včetně nezpracovaných bajtů, řetězcových dat a informací o struktuře a obsahu QR kódu.

jsqr je aktivně udržovaná knihovna a příspěvky do projektu jsou vítány. Je licencována pod licencí Apache-2.0.

### 6.2.2 @zxing/library

@zxing/library [13] je open source, multiformátová knihovna pro zpracování 1D/2D čárových kódů implementovaná v Javě a má porty do jiných jazyků. Je součástí projektu ZXing. Tato knihovna je všestranná a podporuje širokou škálu formátů čárových kódů, včetně QR kodů.

Knihovna momentálně již není v aktivním vývoji a neexistují žádné plány na budoucí změny. Stále se však jedná o spolehlivý nástroj pro čtení a generování kódů.

Ačkoli je kompatibilní s moderními prohlížeči, starší prohlížeče mohou vyžadovat polyfilly pro plnou funkčnost.

[2] `https://npmtrends.com/@zxing/library-vs-html5-qrcode-vs-jsqr-vs-qr-scanner-vs-react-qr-reader`

Instalace je jednoduchá, pomocí npm nebo yarn, a knihovna nabízí několik způsobů, jak integrovat skenování čárových kódů do aplikací, ať už v prohlížeči, nebo v rámci Node.js. Je to všestranný nástroj vhodný pro různé aplikace včetně skenování a generování čárových kódů.

### 6.2.3 react-qr-reader

Knihovna react-qr-reader [14] je komponenta React určená pro skenování QR kódů ve webových aplikacích pomocí knihovny Zxing. Poskytuje přímý způsob, jak integrovat funkci skenování QR kódu do aplikací React, zeja pro použití ve webových prohlížečích.

Knihovna byla testována a je kompatibilní s hlavními prohlížeči, jako jsou Chrome, Firefox a Safari. K dispozici jsou pokyny pro zajištění kompatibility knihovny se staršími prohlížeči.

Knihovna se používá importem komponenty QrReader. Komponenta QrReader obsahuje několik vlastností pro konfiguraci, včetně omezení kamery, obslužných rutin událostí pro výsledky skenování a různých možností stylů pro prvky videa a kontejneru.

Je to open-source projekt, který vítá příspěvky od komunity.

Tato knihovna je vynikající volbou pro vývojáře, kteří chtějí přidat možnosti skenování QR kódů do svých webových aplikací založených na Reactu a nabízí flexibilitu a snadné použití.

### 6.2.4 html5-qrcode

Knihovna html5-qrcode [15] využívá HTML5 getUserMedia API pro přístup k fotoaparátu zařízení pro skenování QR kódu v reálném čase. Je vysoce citlivý a může pracovat s různými rozlišeními fotoaparátu. Vývojáři mohou definovat oblast skenování v pohledu kamery, což umožňuje cílenější a efektivnější skenování.

Knihovna je navržena tak, aby byla kompatibilní s více prohlížeči a zařízeními, takže je všestranná pro širokou škálu webových aplikací. Poskytuje jednoduché a přímočaré rozhraní API, které usnadňuje integraci do stávajících webových projektů s minimální konfigurací. Vývojáři mají flexibilitu přizpůsobit vzhled a chování skeneru tak, aby odpovídal designu jejich aplikace.

V případech, kdy skenování v reálném čase není možné, nabízí knihovna možnost skenovat QR kódy ze statických obrázků.

Celkově knihovna html5-qrcode nabízí pohodlné, efektivní a všestranné řešení pro integraci funkce skenování QR kódu do webových aplikací.

### 6.2.5 qr-scanner

Knihovna poskytuje robustní a efektivní způsob integrace funkce skenování QR kódu do webových aplikací. Využívá moderní webové technologie, které umožňují bezproblémové dekódování QR kódů z různých zdrojů.

Je optimalizována tak, aby efektivně zvládala různá rozlišení kamery a světelné podmínky. Dokáže automaticky vybrat nejvhodnější kameru na zařízení, což je užitečné zeja v zařízeních s více kamerami. Kromě skenování v reálném čase knihovna také podporuje skenování QR kódů z obrázků.

Implementuje optimalizované algoritmy pro rychlé a přesné dekódování QR kódu, snižuje latenci a zlepšuje uživatelský zážitek.

Nabízí přímočaré rozhraní API, které zjednodušuje proces integrace skenování QR kódu do webových projektů. Navržena je tak, aby fungovala v celé řadě prohlížečů.

Knihovna QR-Scanner [16] je ideálním řešením pro vývojáře, kteří chtějí do svých webových aplikací začlenit efektivní a spolehlivé skenování QR kódu se zaměřením na výkon a uživatelskou zkušenost.

### 6.2.6 Porovnání knihoven pro skenování QR kódů

V teto práce jsou knihovny pro snímání kódů posuzovány především z hlediska jejich použití pro skenování zboží ve skladu. Předpokládá se, že ve formátu QR kódu je zakódován následující řetězec `'inventory-track':{productID}`. Proto se předpokládá, že QR kódy jsou stejného typu.

Během testování knihoven byl například použit řetězec: inventory-track:baa8bdf7--c6a3-4a46-9b69-d1253f7a3f1b.

Protože celá aplikace předpokládá použití webové kamery, je nejvhodnějším zařízením pro tento účel telefon. Proto byly všechny testy provedeny na dvou zařízeních: POCOX6 a iPhone15. Níže je uvedeno srovnání knihoven podle parametrů:

- skenování v zařízeních se systémy iOS a Android, tj. možnost spustit je v Google Chrome a Safari.
- Doba skenování stejných QR kódů.
- Jejich schopnost vyhledávat poškozené kódy.
- Skenování QR kódu za špatných světelných podmínek.

**Naskenujte QR kódy v Google Chrome, Safari.**

Přesnost rozpoznávání kódu je klíčovým faktorem při skenování kódů pomocí webové kamery. Srovnávací analýza 6.1 se zaměřuje na to, jak úspěšně jsou QR kódy rozpoznávány v různých prohlížečích. Pro každý prohlížeč bylo provedeno 100 skenů kódu v jednotlivých podmínkách, jako je vzdálenost, osvětlení, poloha kamery vůči kódu.

| knihovna | Chrome | Safari |
|---|---|---|
| jsqr | 100 % | 100 % |
| @zxi ng/library | 100 % | 100 % |
| react-qr-reader | 100 % | 100 % |
| html5-qrcode | 100 % | 100 % |

**Tabulka 6.1.** Výsledky skenování QR kódů v různých prohlížečích.

**Průměrná doba skenování QR kódu.**

Dalším důležitým parametrem je rychlost rozpoznání kódu. V prohlížeči Chrome bylo naskenováno 100 různých kódů. Pro každý kód byl zaznamenán čas skenování. Výsledky jsou uvedeny v tabulce níže.

| knihovna | průměrná doba v sekundách |
|---|---|
| jsqr | 0.4 |
| @zxi ng/library | 0.5 |
| react-qr-reader | 0.3 |
| html5-qrcode | 1 |
| qr-scanner | 0.7 |

**Tabulka 6.2.** Průměrná doba skenování kódu.

Podle výsledků 6.2 vykázaly nejkratší dobu skenování knihovny @zxi ng/library a qr-scanner. Nejhůře dopadla knihovna html5-qrcode.

Vzhledem k tomu, že skenování kódů bude prováděno ve skladech, je velká pravděpodobnost, že dojde k poškození kódů např. při přesunu zboží. Proto dalším důležitým parametrem při porovnávání knihoven je, jak dobře jsou schopny pracovat s takovými

| knihovna | úspěšnost |
|---|---|
| jsqr | 100 % |
| @zxi ng/library | 100 % |
| react-qr-reader | 100 % |
| html5-qrcode | 96 % |
| qr-scanner | 98 % |

**Tabulka 6.3.** Výsledky skenování poškozených QR kódů.

QR kódy. K provedení tohoto testu bylo použito 50 kódů s různými typy a stupni poškození.

Podle výsledků 6.3 v si v tomto testu nejlépe vedla knihovna @zxi ng/library. Zbytek vykázal o něco horší výsledky. Knihovna html5-qrcode však byla nejhorší ze všech.

**Skenování QR kódů při špatném osvětlení**

V reálném prostředí skladu je možné, že skenování kódu bude prováděno při slabém osvětlení. Proto dalším důležitým kritériem pro porovnávání knihoven je, jak dobře se vypořádají s rozpoznáváním QR kódů za špatných světelných podmínek.

| knihovna | úspěšnost |
|---|---|
| jsqr | 100 % |
| @zxi ng/library | 100 % |
| react-qr-reader | 100 % |
| html5-qrcode | 100 % |
| qr-scanner | 100 % |

**Tabulka 6.4.** Výsledky skenování kódů při špatném osvětlení.

**Závěr**

Srovnání výsledků testů neukazuje na jednoznačného lídra. Podle provedených testů však některé knihovny vykazovaly o něco lepší výsledky ve srovnání s jinými

V dalších testech, prvním a posledním, vykázaly všechny knihovny vynikající výsledky. Důležité jsou však i další dva testy. Knihovna html5-qrcode vykázala nejvyšší čas pro skenování kódů, nejrychlejšími knihovnami byly jsqr a react-qr-reader. V testu skenování poškozených kódů vykazovaly nejhorší výsledky knihovny html5-qrcode a qr--scanner. Ostatní tři se projevily výborně.

Na základě výsledků tohoto testu však nelze zcela jednoznačně rozhodnout, která z těchto knihoven je nejlepší, proto jsme při výběru knihovny pro použití při implementaci webové aplikace přihlíželi také ke snadné integraci s knihovnou ReactJS.

## 6.3 Závěr

Implementovaná aplikace je umístěna v následujících zdrojích:

- Serverová část aplikace [3].
- Klientská část aplikace [4].
- Aktuální platná aplikace [5].

[3] `https://github.com/AnnaKachmasheva/bp-fel-cvut`
[4] `https://github.com/AnnaKachmasheva/bp-fe-final`
[5] `https://master.d3f81l92tk91tc.amplifyapp.com`

# Kapitola 7
# Testování

Účelem testování je odstranit chyby, které by mohly zhoršit uživatelský zážitek nebo způsobit ztrátu osobních údajů. Na základě případů užití byly napsány testovací scénáře, které ověřují, zda aplikace funguje správně v souladu s funkčními požadavky.

## 7.1 Uživatelské testování

**Testovací scénář 1:**
**Data:** Neregistrovaný uživatel a nepřihlášený uživatel.
**Kroky:**

- Uživatel přejde pomocí tlačítka `Registration` do sekce Registrace uživatelů.
- Uživatel vyplní nezbytné údaje – First name, Last name, Email address, Password.
- Repeat password a dokončí registraci pomocí tlačítka `Create account`.

**Výstup:** Automatické přihlášení uživatele po registraci.
**Testovací scénář 2:**
**Data:** Autorizace.
**Kroky:**

- Uživatel přejde pomocí tlačítka `Log in` do sekce Autorizace uživatelů.
- Uživatel vyplní přihlašovací formulář a stiskne tlačítko `Continue`.

**Výstup:** Automatické přihlášení uživatele po registraci.
**Testovací scénář 3:**
**Data:** Tvorba nového produktu.
**Kroky:**

- Správce přejde do sekce `Storage`.
- Pomocí tlačítka `Add product` otevře modální okno s tvorbou produktu.
- Správce vyplní údaje – Category, State, Name, Description, Image a pomocí tlačítka `Save` uloží nový produkt.

**Výstup:** Produkt je uložen do databáze.
**Testovací scénář 4:**
**Data:** Tvorba nové objednávky.
**Kroky:**

- Správce přejde do sekce `Orders`.
- Pomocí tlačítka `Add order` otevře modální okno vytvoření objednávky.
- Po výběru produktu pro objednávku správce uloží objednávku pomocí tlačítka `Save`.

**Výstup:** Objednávka je uložena.
**Testovací scénář 5:**
**Data:** Zobrazení detailů produktu.
**Kroky:**

- Uživatel přejde do sekce `Storage`.
- Kliknutím na produkt se uživatel dostane na podrobnosti o produktu.

**Výstup:** Uživatel vidí detail produktu.
**Testovací scénář 6:**
**Data:** Změna osobních údajů.
**Kroky:**

- Uživatel přejde do sekce `Profile`.
- Pomocí tlačítka `Edit` uživatel otevře modální okno pro změnu osobních údajů.
- Uživatel změní uložená data – First Name, Last Name, Email address a pomocí tlačítka `Save changes` uloží změnu.

**Výstup:** Nová data jsou uložena do databáze.
**Testovací scénář 7:**
**Data:** Změna hesla profilu.
**Kroky:**

- Uživatel přejde do sekce `Profile`.
- Pomocí tlačítka `Change password` uživatel otevře modální okno pro změnu hesla.
- Uživatel musí dvakrát zadat staré a dvakrát nové heslo a pomocí tlačítka `Save changes` změny uložit.

**Výstup:** Nová data jsou uložena do databáze.
**Testovací scénář 8:**
**Data:** Mazání profilu uživatele.
**Kroky:**

- Uživatel přejde do sekce `Profile`.
- Pomocí tlačítka `Delete account` uživatel smaže svůj profil a bude přesměrován na úvodní stránku.

**Výstup:** V databázi uživatel získal příznak `isDeleted` = true.
**Testovací scénář 9:**
**Data:** Správa objednávek.
**Kroky:**

- Uživatel přejde do sekce `Orders`.
- Uživatel vybere objednávku.
- Uživatel spustí zpracování objednávky pomocí tlačítka `Start order`.
- Uživatel spustí postup vytváření objednávky pomocí tlačítka `Start scan`.
- Postupně vybere jednotlivé produkty ze seznamu a naskenuje QR kód pomocí fotoaparátu.
- Po naskenování všech produktů v objednávce dokončí objednávku pomocí tlačítka `Complete`.

**Výstup:** Stav objednávky `Completed`, stav produktů dokončené objednávky `Delivered`.

### 7.1.1 Cílová skupina

Vzhledem k tomu, že stávající řešení jsou často popisována jako složitá a nepohodlná pro používání, bylo rozhodnuto vybrat skupinu lidí, kteří nemají specializované dovednosti a kterým oblast správy zboží, plánování, vydávání a zpracování objednavek není blízká.

### 7.1.2 Průběh testování

Vybraná skupina osob použila k testování napsané testovací scénáře. Po provedení testování aplikace byla získána zpětná vazba, která umožnila zlepšit funkčnost aplikace a opustit neoptimální řešení.

### 7.1.3 Výsledky testování

Výsledkem tohoto testování bylo zlepšení výkonu aplikace. Bylo odstraněno několik kritických chyb a řada nedostatků souvisejících s návrhem uživatelského rozhraní.

# Kapitola 8
# Závěr

V rámci této bakalářské práce byla prokázána analýza stávajících řešení systémů pro řízení zásob ve skladu a zkoumání možných technologických řešení pro použití skenování QR kódů zboží pomocí webové kamery. Na základě zadání bakalářské práce a těchto zjištění byl vypracován návrh aplikace. Byly definovány funkční a nefunkční požadavky, scénáře použití pro klienta, správce a neoprávněného uživatele, datový model a procrsní diagramy.

Na základě analýzy a s ohledem na splnění požadavků na funkčnost od potenciálních uživatelů systému byla provedena implementace nabídky řešení od Lofi po HiFi prototyp.

Provedli jsme analýzu moderních technologií pro implementaci systému správy zboží a vybrali nejvhodnější z nich, které splňovaly požadavky na bezpečnost, spolehlivost a snadné použití. Pro klientskou část byl zvolen jazyk JavaScript a knihovna ReactJS, pro serverovou část jazyk Java a framework Spring Boot. Jako datové úložiště byla použita relační databáze PostgreSQL, protože je to jedna z podmínek zadání. Pro implementaci vstupních bodů bylo použito otevřené API. Pro implementaci aplikace byl použit princip API First. Backend aplikace byl nasazen na cloudové platformě Azure pomocí služby Azure Spring App. Klientská část byla nasazena na cloudové platformě AWS pomocí služby AWS Amplify.

Charakteristickým rysem systému pro správu zboží je využití QR kódů a webové kamery pro rychlé vyhledávání informací o výrobku a provádění objednávek. Proto byly prozkoumány, otestovány a porovnány nejčastěji stahované open-source JS knihovny pro skenování kódů. Testy knihoven však neukázaly žádné výsledky, které by se od sebe výrazně lišily. Pravděpodobně je to způsobeno malým počtem testů, který činil 100 pro každou knihovnu pro každý test. Proto byla pro implementaci systému vybrána knihovna, která vykazovala nejlepší výsledky a byla snadno integrovatelná s ReactJS.

Implementovaná aplikace byla otestována uživatelskými testy. Tyto testy umožnily odstranit velké množství chyb a nedostatků.

Implementovaná aplikace poskytuje spíše omezenou funkčnost a mohla by být dále vylepšena pro zlepšení uživatelského zážitku:

- Rozšíření možnosti přidávat nové kategorie produktů.
- Přidání možnosti nahrát více obrázků pro jeden produkt.
- Rozšíření údajů zachycených ve statistikách, které poskytnou více informací o produktech a objednávkách.
- Zavedení mobilní aplikace.

Implementace těchto a dalších možných rozšíření by měla být provedena po důkladné analýze systému a podle obchodních požadavků uživatelů.

# Literatura


[1] Rehaul Awati | Ivy Wigmore. *Monolithic architecture*. 2024. https://www.techtarget.com/whatis/definition/monolithic-architecture.

[2] Oracle. *Java*. 2024. https://www.java.com/en/download/help/whatis_java.html.

[3] Broadcom. *Spring boot*. 2024. https://spring.io/projects/spring-boot.

[4] The PostgreSQL Global Development Group. *PostgreSQL*. 2024. https://www.postgresql.org/.

[5] Apache Software Foundation. *ACID*. 2024. https://www.databricks.com/glossary/acid-transactions##:~:text=ACID%20is%20an%20acronym%20that,operations%20are%20called%20transactional%20systems..

[6] Microsoft. *Azure Spring Apps*. 2024. https://azure.microsoft.com/en-us/products/spring-apps.

[7] Meta Open Source. *React*. 2024. https://react.dev/learn.

[8] Inc. Amazon Web Services. *AWS Amplify*. 2024. https://aws.amazon.com/amplify/.

[9] Red Hat. *Hibernate ORM*. 2024. https://hibernate.org/orm/.

[10] Spring Boot Rest. *Broadcom*. 2024. https://spring.io/guides/tutorials/rest.

[11] Eric Cabrel Tiogo. *JWT authentication*. 2024. https://medium.com/@tericcabrel/implement-jwt-authentication-in-a-spring-boot-3-application-5839e4fd8fac.

[12] Cosmo Wolfe. *jsqr*. 2024. https://github.com/cozmo/jsQR.

[13] ZXing Project. *@zxing/library*. 2024. https://github.com/zxing/zxing.

[14] Yudiel Curbelo. *react-qr-reader*. 2024. https://github.com/yudielcurbelo/react-qr-scanner.

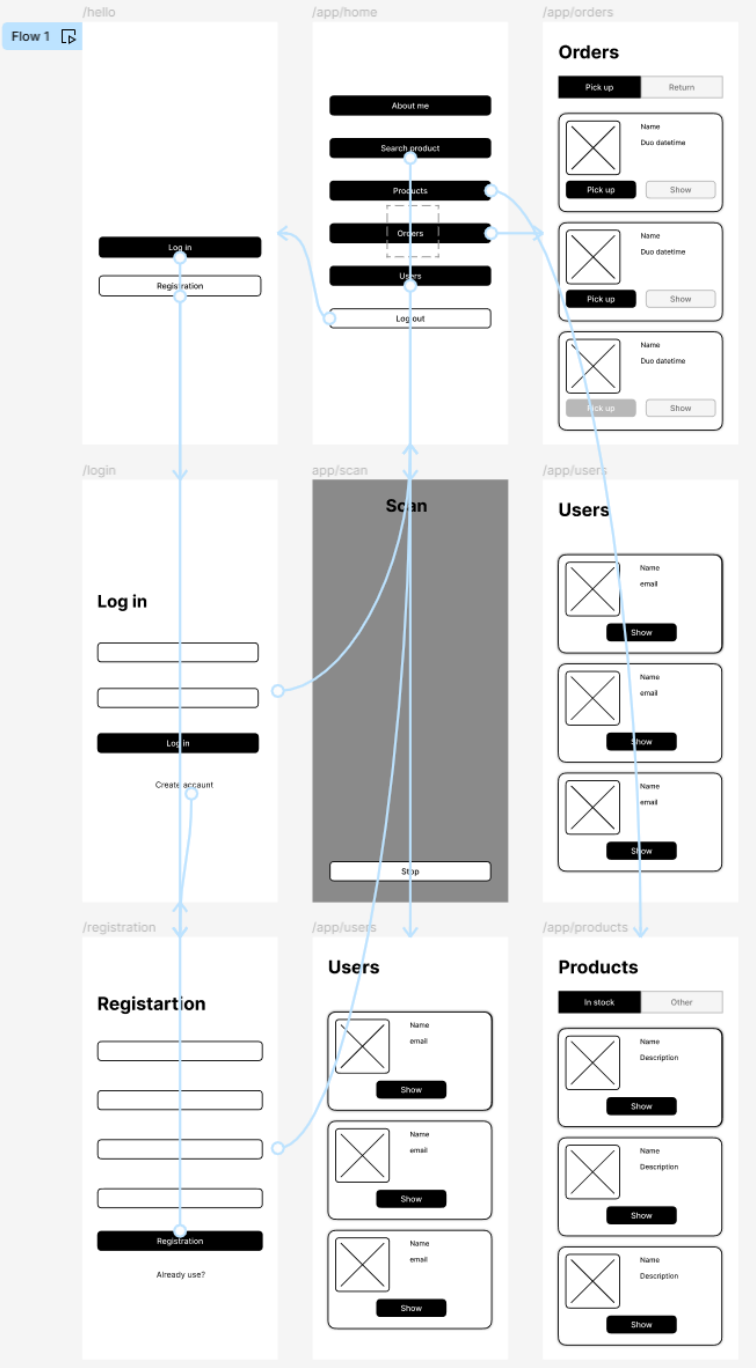
[15] Minhaz. *html5-qrcode*. 2024. https://github.com/mebjas/html5-qrcode.

[16] Nimiq Blockchain Ecosystem. *qr-scanner*. 2024. https://github.com/nimiq/qr-scanner.

# Příloha A
# Seznam použitých zkratek

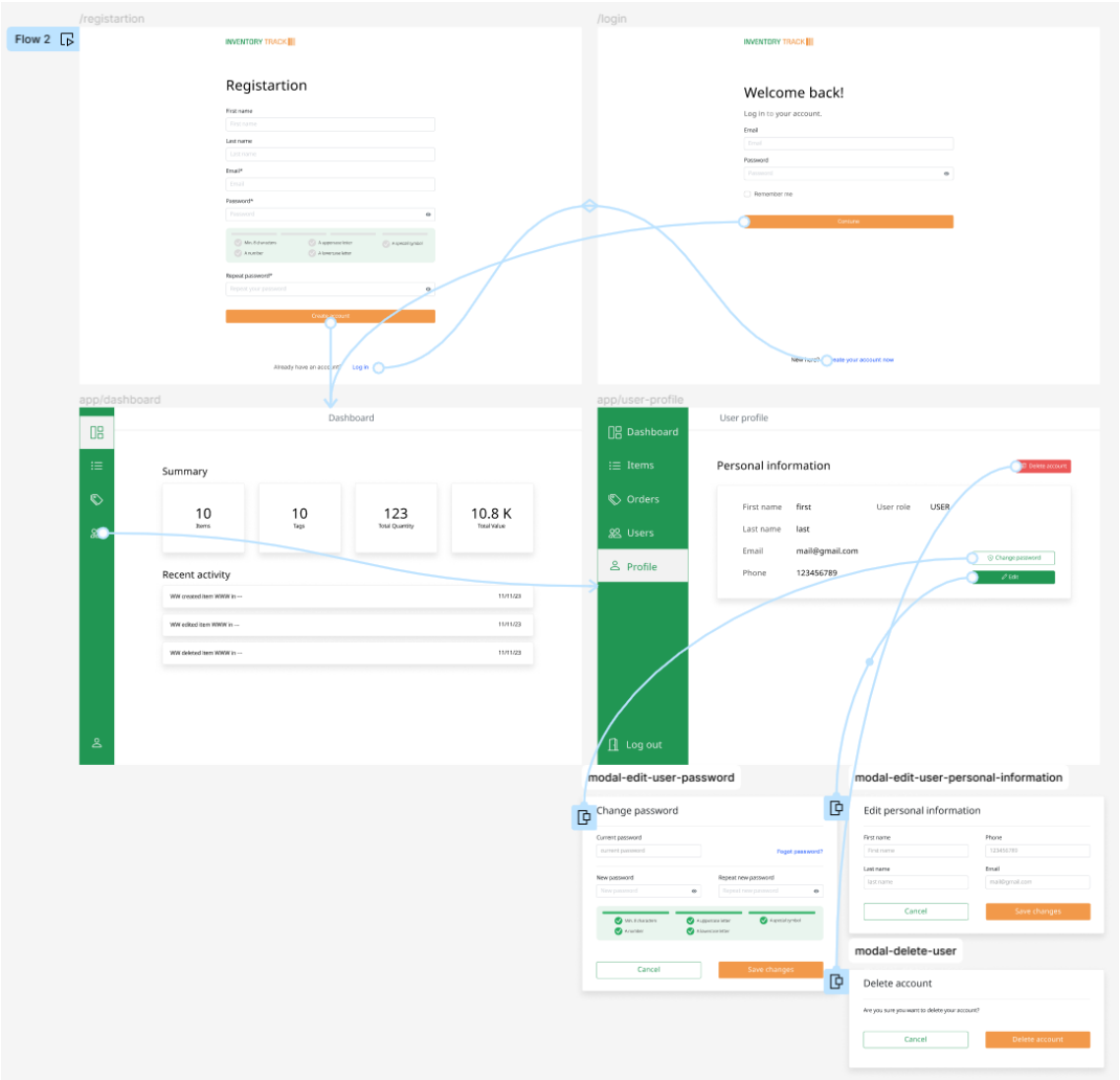
| | |
|---|---|
| 1D/2D | One dimensional/Two dimensional |
| ACID | Atomicity, Consistency, Isolation, and Durability |
| AWS | Amazon Web Services |
| API | Application Programming Interface |
| BigInt | Big Integer |
| CI/CD | Continuous Integration/Continuous Delivery |
| CRUD | Create, Read, Update and Delete |
| CRM | Customer Relationship Management |
| CSV | Comma-separated values |
| DAO | Data Access Object |
| ERP | Enterprise Resource Planning |
| FR | Funkční požadavk |
| Hi-fi | High fidelity |
| HTML5 | HyperText Markup Language |
| HTTP | Hypertext Transfer Protocol |
| IaaS | Infrastructure as a Service |
| iOS | iPhone Operating System |
| JSON | JavaScript Object Notation |
| JPA | Java Persistence API |
| JS | JavaScript |
| JWT | JSON Web Token |
| Lo-fi | Low fidelity |
| MIT | Massachusetts Institute of Technology |
| NFR | Nefunkční požadavek |
| PaaS | Platform as a Service |
| REST API | Representational State Transfer Application Programming Interface |
| RFID | Radio-frequency identification |
| SQL | Structured Query Language |
| UC | Use Case |

# Příloha B
## Příklad Lo-fi prototypu

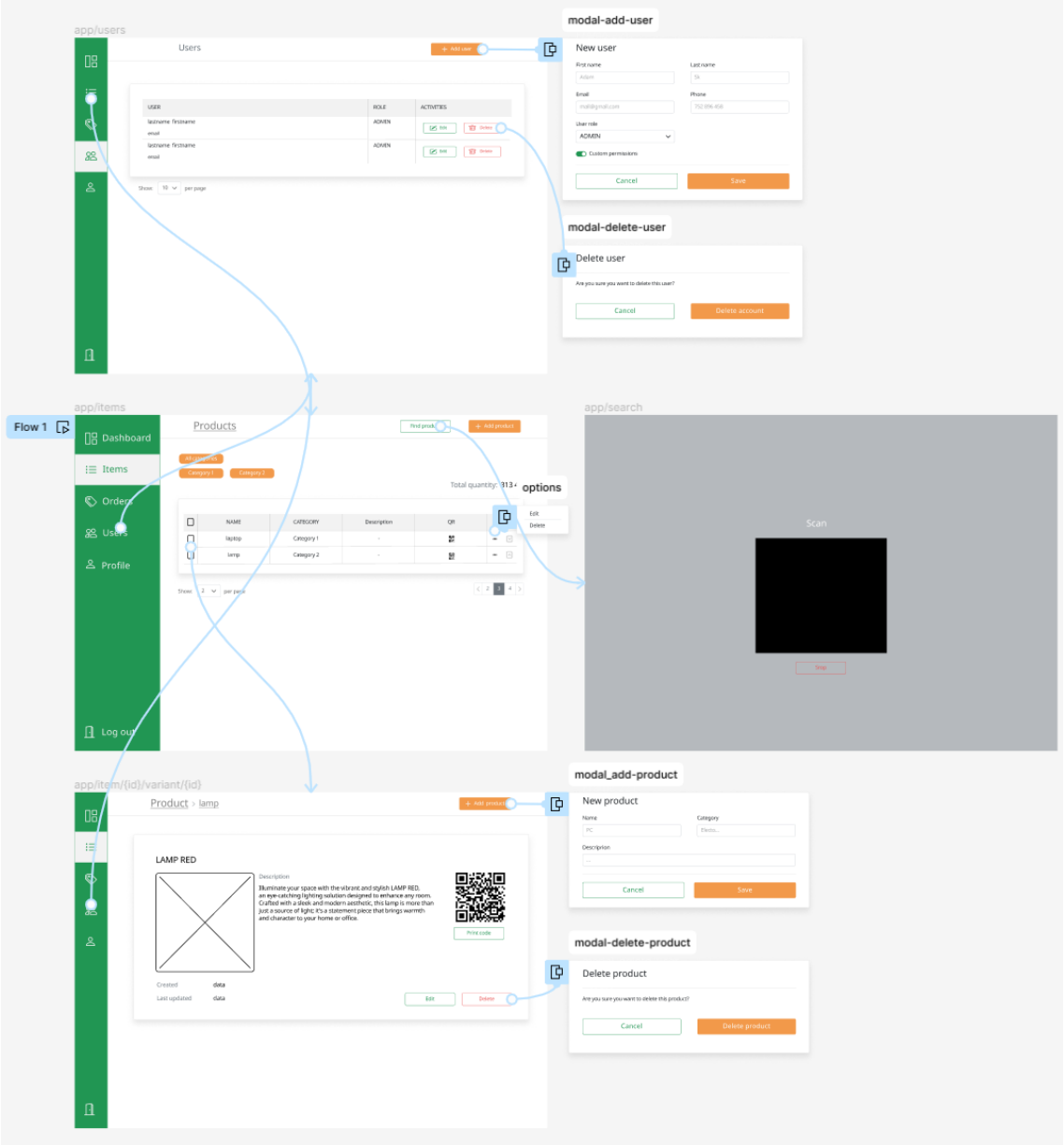



**Obrázek B.1.** Prototyp Lo-fi.

# Příloha C
## Příklad Hi-fi prototypu




**Obrázek C.2.** Sekce autorizace, registrace, dashboard a profile (Prototyp Hi-fi).




**Obrázek C.3.** Sekce produktů a uživatelů (Prototyp Hi-fi).