FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Bakalářská práce

Elektronický sběr základních dat pro průběžnou evidenci odpadů

Studijní program: B0613A140005 Informační technologie
Studijní obor: Aplikovaná informatika

Autor práce: **Nikita Tashchilin**
Vedoucí práce: Ing. Mojmír Volf
Ústav nových technologií a aplikované informatiky

Liberec 2023

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Zadání bakalářské práce

Elektronický sběr základních dat pro průběžnou evidenci odpadů

Jméno a příjmení:	**Nikita Tashchilin**
Osobní číslo:	M20000019
Studijní program:	B0613A140005 Informační technologie
Specializace:	Aplikovaná informatika
Zadávající katedra:	Ústav nových technologií a aplikované informatiky
Akademický rok:	2022/2023

Zásady pro vypracování:

1. Seznamte se s problematikou tvorby progresivních webových aplikací a problematikou průběžné evidence odpadů podle aktuálně platné legislativy ČR.
2. Na základě získaných znalostí navrhněte způsob elektronického sběru základních dat o odpadech vznikajících zejména ve zdravotnických zařízení pro následnou tvorbu zákonné průběžné evidence.
3. Vytvořený návrh prakticky implementujte a naprogramujte on-line systém pro sběr základních údajů souvisejících s průběžnou evidencí odpadů včetně příslušné uživatelské a programátorské dokumentace.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	30 – 40 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	Čeština

Seznam odborné literatury:

[1] Zákon č. 541/2020 Sb. o odpadech. In: *Sbírka zákonů*. 23. 12. 2020. ISSN 1211-1244.

[2] HUME, Dean Alan and OSMANI, Addy. *Progressive web apps*. Shelter Island, NY : Manning Publications, 2018. ISBN 1617294586.

[3] ATER, Tal. *Building progressive web apps: Bringing the power of native to the browser*. Sebastopol, CA : O'Reilly Media, 2017. ISBN 9781491961650.

Vedoucí práce:	Ing. Mojmír Volf Ústav nových technologií a aplikované informatiky
Datum zadání práce:	12. října 2022
Předpokládaný termín odevzdání:	22. května 2023

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

Ing. Josef Novák, Ph.D.
vedoucí ústavu

V Liberci dne 19. října 2022

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

17. 5. 2023 Nikita Tashchilin

ELEKTRONICKÝ SBĚR ZÁKLADNÍCH DAT PRO PRŮBĚŽNOU EVIDENCI ODPADŮ

ABSTRAKT

Tato bakalářská práce si klade za cíl vyvinout aplikaci pro vytváření datových podkladů průběžné evidence odpadů v nemocnicích a ve zdravotnických zařízeních s možností následného exportu dat do specializovaného programu IS ENVITA (dříve EVI 8) pro automatizovanou tvorbu kompletní průběžné evidence odpadů a Ročního resp. Souhrnného hlášení o produkci a nakládání s odpady.

Klíčová slova: webová aplikace, evidence odpadů

ELECTRONIC COLLECTION OF BASIC DATA FOR CONTINUOUS WASTE RECORDING

ABSTRACT

This bachelor thesis aims to create an application for creating data records of ongoing waste management in hospitals and medical facilities with the option to export data to the specialized IS ENVITA software (formerly EVI 8) for automated creation of a complete waste management record and an Annual or Summary report on waste production and handling.

Keywords: web application, waste records

PODĚKOVÁNÍ

Rád bych poděkoval panu Petru Grusmanovi a dalším zaměstnancům společnosti INISOFT s.r.o. za to, že mi poskytli zázemí a svými připomínkami přispěli ke vzniku tohoto díla. V neposlední řadě bych chtěl poděkovat panu Ing. Mojmíru Volfovi za vedení této práce.

OBSAH

Úvod **11**

Cíl práce **12**

1 Základní požadavky na elektronický sběr dat **13**
1.1 Rozsah dat pro průběžnou evidenci odpadů 13
1.2 Prostředí pro sběr dat (lokálně, Intranet, Internet) 14
1.3 Rozhraní pro předání vytvořené evidence 14

2 Alternativy **15**

3 Návrh řešení **16**
3.1 Databáze 17
3.2 Backend 17
3.2.1 Framework NestJS 17
3.2.2 TypeScript 18
3.2.3 API 19
3.3 Uživatelské rozhraní 19
3.3.1 React 19
3.3.2 Next.js 20
3.3.3 Shrnutí technologií 20
3.3.4 Jednoduchost použití 21
3.4 Infrastruktura 22
3.4.1 Docker. Je správnou volbou? 22
3.4.2 Docker Compose 23
3.4.3 Nginx 24
3.5 Export 24
3.5.1 XML pro ostatní aplikace 24
3.5.2 XLSX pro uživatele 25

4 Realizace řešeni **26**
4.1 Serverová část 26
4.1.1 Struktura 26
4.1.2 Export 27
4.1.3 Databáze 28
4.1.4 Zabezpečení 28

4.2 Captcha 30
4.3 Webové rozhraní 31
4.3.1 Technologie a knihovny 31
4.3.2 Mobilní aplikace 32
4.4 Infrastruktura 33
4.4.1 Praktické použiti Dockeru 33
4.4.2 Nginx 34
4.4.3 Verzování kódu 34
4.4.4 Architektura a propojení jednotlivých části aplikace . . . 35
4.5 Testovaní 36
4.5.1 Automatizované testování 36
4.5.2 Prostředí pro ruční testování 36
4.6 Možnosti dalšího využití a rozšíření aplikace 36

Závěr **37**

Literatura **38**

A Přílohy **40**
A.1 Ukázka exportu 40
A.2 Ukázka uživatelského rozhraní 42
A.3 Odkaz na zdrojové kódy 44

SEZNAM OBRÁZKŮ

3.1 Use-case diagram 16
3.2 Docker oproti VM 23

4.1 Struktura backend části 27
4.2 Finální struktura databáze 28
4.3 Základní konfigurační soubor 34
4.4 Architektura aplikace 35

SEZNAM ZKRATEK

JISPEN	Jednoduchý informační systém pro sběr dat o odpadech v nemocnicích a dalších zdravotnických zařízeních
MVP	Minimum viable product
ORP	Obec s rozšířenou působností
ZÚJ	Základní územní jednotka
PSČ	Poštovní směrovací číslo
PWA	Progressive Web App
XML	eXtensible Markup Language
XSD	XML Schema Definition
CRUD	Create, Read, Update, Delete
BE	Backend
DI	Dependency Injection
UI	User Interface
UX	User Experience
DOM	Document Object Model
SSR	Server-Side Rendering
SEO	Search Engine Optimization
VM	Virtual Machine
TTL	Time-To-Live
JWT	JSON Web Token
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP Secure
MITM	Man-in-the-Middle
ORM	Object-Relational Mapping
API	Application Programming Interface
SWR	Stale-While-Revalidate
PWA	Progressive Web App

ÚVOD

Povinnost vedení průběžné evidence odpadů vyplývá z aktuálně platného zákona o odpadech č. 541/2020 Sb. Je nutné ji dodržovat pro všechny právnické a fyzické osoby oprávněné k podnikání [1] a představuje pro ně administrativní zátěž.

Proto v rámci ročníkového projektu byla vyvinuta MVP aplikace JISPEN určená pro zaměstnance nemocnic a zdravotnických zařízení, kteří nemají hlubší znalost legislativních požadavků v oblasti odpadového hospodářství.

V aplikaci si uživatelé zaznamenávají informace o produkci odpadů pomocí předem připravených šablon, které omezují rozsah evidovaných údajů pouze na nezbytné informace a usnadňují zápis. Tyto údaje se poté elektronicky využívají k vytvoření kompletní průběžné evidence odpadů v programech jako je např. IS ENVITA (dříve EVI 8).

V důsledku bezproblémového testování MVP aplikace JISPEN a pozitivního ohlasu jak uživatelů, tak u zákazníka, se rozhodlo o pokračování v dalším rozvoji a vylepšování aplikace. Cílem je dále usnadňovat a zefektivňovat proces vedení evidence odpadů a tím pomoci snižovat administrativní zátěž všem dotčeným subjektům.

CÍL PRÁCE

Cílem této práce je dále rozvíjet a vylepšovat MVP aplikaci JISPEN, která byla vyvinuta pro usnadnění procesu vedení průběžné evidence odpadů v nemocnicích a zdravotnických zařízeních. Mezi hlavní cíle práce patří následující body:

- Rozšíření funkcionalit aplikace - cílem je přidat nové funkce a možnosti, které nabídnou uživatelům aplikace efektivněji zaznamenávat a spravovat informace o produkci odpadů.
- Zlepšení uživatelského rozhraní - cílem je vylepšit uživatelské rozhraní aplikace tak, aby bylo co nejpřehlednější a nejjednodušší na používání, a tím uživatelům umožnilo snadno a rychle pracovat s aplikací.
- Integrace s dalšími informačními systémy - cílem je zajistit propojení aplikace JISPEN s dalšími informačními systémy, které se v nemocnicích a zdravotnických zařízeních používají, aby se usnadnilo a zefektivnilo využívání dat o produkci odpadů.
- Snižování administrativní zátěže - cílem je přispět ke snižování administrativní zátěže subjektům povinných vést průběžnou evidenci odpadů tím, že se jim usnadní a zefektivní tato činnost právě pomocí aplikace JISPEN.

Hlavním cílem této bakalářské práce je tedy přispět ke zefektivnění a zjednodušení vedení průběžné evidence odpadů v nemocnicích a zdravotnických zařízeních a tím usnadnit plnění povinností dle platného zákona o odpadech.

1 ZÁKLADNÍ POŽADAVKY NA ELEKTRONICKÝ SBĚR DAT

Během vývoje MVP v rámci ročníkového projektu byly zohledněny tyto požadavky:

- Dostupnost aplikace pro koncového uživatele, bez nutnosti instalace podpůrných souborů, systémů či knihoven.
- Aplikace je bezplatná pro koncového uživatele.
- Možnost ručního založení základních údajů pro tvorbu evidence, jako jsou údaje o provozovně, povolené kódy odpadů a nakládání, seznam partnerských subjektů.
- Aplikace obsahuje číselníky pro snadné nastavení šablony, jako ZÚJ, PSČ, Katalog odpadů a kódy nakládání.
- Editační formulář pro vlastní vedení evidence.

Nicméně pro usnadňovaní a zefektivňovaní procesu vedení evidence odpadů bylo potřeba tyto požadavky rozšířit.

1.1 ROZSAH DAT PRO PRŮBĚŽNOU EVIDENCI ODPADŮ

Aplikace by měla umět pracovat s následujícími datovými položkami:

- Kód druhu odpadu (šestimístný/osmimístný kód, kategorie, název)
- Kód nakládání s odpadem
- Identifikace producenta odpadu (provozovna)
- Identifikace předávající osoby/příjemce odpadu (oprávněna osoba)
- Množství odpadu (tuny/kilogramy)
- Datum produkce/předání/převzetí odpadu

Tyto informace jsou důležité pro dodržování legislativních povinností týkajících se vedení zákonné průběžné evidence odpadů a pro získání přehledu o množství a druhu odpadů.

1.2 Prostředí pro sběr dat (lokálně, intranet, internet)

Určení vhodného prostředí pro sběr dat je klíčovým faktorem pro úspěšné používání aplikace. Vzhledem k tomu, že aplikace by měla být dostupná a použitelná na široké škále zařízení, včetně lokálních počítačů, intranetů (soukromé počítačové sítě určené pro vnitřní potřeby organizace [2]) a internetu (veřejně přístupný prostor), je nutné zajistit, aby byla aplikace navržena tak, aby byla kompatibilní s různými operačními systémy a konfiguracemi sítě. Tento přístup umožní uživatelům snadné používání aplikace bez ohledu na to, kde se právě nacházejí. To vše přispívá k větší efektivitě a produktivitě práce s aplikací a také k větší spokojenosti uživatelů.

1.3 Rozhraní pro předání vytvořené evidence

Aby byla aplikace co nejužitečnější pro uživatele, je důležité, aby byla vybavena kvalitním rozhraním pro přenos vytvořených dat. Toto rozhraní by mělo být navrženo s ohledem na jednoduchost, intuitivnost a efektivitu, aby uživatelé mohli snadno a rychle předávat informace do různých systémů, nebo externím subjektům, které potřebují tyto informace ke své práci.

Pro účely přenosu dat mezi různými systémy je nezbytné použít formát, který je široce používán v oblasti odpadového hospodářství a zároveň zajistí snadnou a bezproblémovou kompatibilitu dat. Vzhledem k tomu, že systém IS ENVITA od společnosti INISOFT s.r.o. je nejrozšířenější systém pro vedení kompletní průběžné evidence odpadů v ČR, bude použit XML formát s předem definovaným XSD popisem, který je v IS ENVITA podporován.

2 ALTERNATIVY

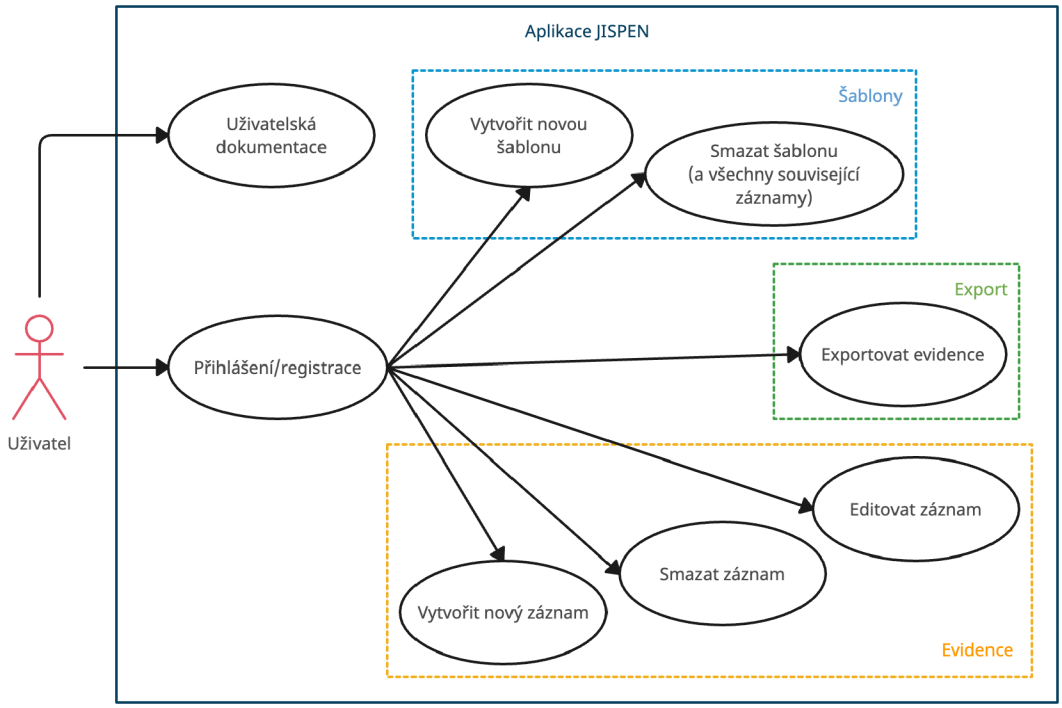
V současné době na trhu v ČR existují pouze větší programy, jako ENVITA a EVI 8, které jsou navrženy pro kompletní tvorbu průběžné evidence odpadů a elektronického sestavení Ročního resp. Souhrnného hlášení o odpadech. Pro menší organizace a zdravotnická zařízení resp. koncové uživatele jsou často příliš složité, drahé a překračují jejich potřeby.

Aplikace JISPEN by měla nabízet alternativní bezplatné řešení pro správné vytváření podkladů k sestavení evidence odpadů, čímž sníží administrativní zátěž zodpovědným osobám, poskytne nástroj všem uživatelům z řad zdravotnického personálu a zajistí elektronickou výměnu informací o odpadech z místa jejich vzniku až po následné zpracování v kompletní průběžné evidenci odpadů za každou provozovnu. Mezi hlavní výhody aplikace bude patřit především přístup z jakéhokoliv místa a zařízení (mobilní telefon, tablet, počítač). JISPEN bude navržen tak, aby umožnil uživatelům efektivnější správu odpadů, přičemž musí zůstat přehledný a snadno použitelný pro všechny uživatele.

3 NÁVRH ŘEŠENÍ

V následující kapitole se zaměřím na návrh aplikace JISPEN. Pro tuto bakalářskou práci jsem použil jako základ svůj ročníkový projekt "Prototyp aplikace JISPEN", při jehož navrhování bylo rozhodnuto, že aplikace bude ve formě webové aplikace, protože tento typ aplikace nevyžaduje instalaci a umožňuje přístup z jakéhokoliv zařízení bez poplatků, což bylo pro zákazníka velmi důležité.

Pro lepší porozumění funkčnosti aplikace byl vytvořen use-case diagram, který popisuje interakce uživatele s aplikací. Díky tomu bylo možné lépe porozumět potřebám uživatelů a navrhnout aplikaci tak, aby jim co nejvíce vyhovovala.



Obrázek 3.1: Use-case diagram

3.1 DATABÁZE

Při vývoji aplikaci JISPEN jsem nadále používal databázový systém PostgreSQL, který jsem vybral v rámci ročníkového projektu. Tento databázový systém se ukázal jako optimální řešení pro výkon, spolehlivost a bezpečnostní požadavky aplikace. Je open-source a nabízí širokou škálu funkcí, což z něj dělá jeden z nejpoužívanějších a nejvýkonnějších relačních databázových systémů [3].

Pro snadné zpracování dat v aplikaci a zajištění bezpečnosti manipulace s daty používám ORM framework, konkrétně TypeORM. Tento framework umožňuje použití objektově-relačního mapování pro databáze, což usnadňuje vývoj a údržbu aplikací. Pro efektivní manipulaci s daty jsou v aplikaci implementovány CRUD operace. Například **Create** se používá k vytvoření nové šablony nebo záznamu evidence odpadu, **Read** zajistí zobrazení dat uživateli, **Update** aktualizuje záznam a **Delete** se používá pro odstranění šablony nebo evidenci.

Pro správu databáze a udržení konzistence datových struktur používám migrace. Migrace umožňují jednoduchou a bezpečnou aktualizaci databází při zajištění struktury databáze a souladu dat s definovaným schématem. Migrace probíhá automaticky a pouze před spuštěním aplikace v produkčním prostředí. TypeORM umožňuje snadno provádět aktualizace databáze na základě změn entit bez ručního zásahu. Tento proces je plně automatizovaný a usnadňuje správu a údržbu databáze. Díky migracím bylo umožněno snadno a rychle aktualizovat databázi v produkci.

3.2 BACKEND

V této kapitole se budeme věnovat backendu aplikace JISPEN. Backend je část aplikace, která se stará o zpracování a ukládání dat, o aplikační logiku a komunikaci s externími systémy. V rámci této kapitoly se zaměříme na framework NestJS, který byl zvolen pro vývoj backendu aplikace JISPEN.

3.2.1 Framework NestJS

NestJS je v současné době jedním z nejpopulárnějších JavaScriptových backend frameworků, postavených na Node.js. Node.js je JavaScriptový runtime postavený na platformě V8 od společnosti Google, což je výkonný JavaScriptový engine, který byl původně vyvinut pro použití v prohlížeči Google Chrome a je v současné době používán i v dalších projektech, jako například v Node.js nebo Deno. Výhodou použití V8 v Node.js je především

jeho rychlost a výkon, což umožňuje spouštění a bezproblémový běh náročných aplikací a služeb.

Jednou z klíčových vlastností Node.js je jeho event-driven architektura, která umožňuje asynchronní operace a neblokující zpracování dat. Node.js používá event-loop, což je způsob zpracovávání událostí, při kterém Node.js neustále čeká na nové události, které mohou nastat (například nový požadavek k API), a poté spouští související kód pro zpracování [4]. Tento přístup umožňuje Node.js reagovat na vysoký počet požadavků a paralelně provádět několik operací najednou, aniž by došlo ke zpomalení serveru.

V neposlední řadě, NestJS poskytuje komunitu s mnoha užitečnými rozšířeními a nástroji, jako například Swagger UI pro dokumentaci API, což usnadňuje vývoj a údržbu aplikací.

NestJS byl zvolen pro vývoj MVP aplikace JISPEN a nadále se používá kvůli jeho mnoha výhodám oproti konkurenčním BE knihovnám v Node.js, jako jsou například Express nebo Fastify. NestJS nabízí výbornou podporu DI (Dependency Injection), modulů a middleware, což usnadňuje vývoj a údržbu API. Navíc, díky jednoduché struktuře založené na principu MVC je možné vytvářet stabilní, škálovatelné a snadno testovatelné aplikace.

3.2.2 TypeScript

TypeScript je nadstavbou programovacího jazyka JavaScript, která přináší možnost statického typování a mnoho dalších vylepšení. Díky TypeScriptu je možné odhalovat chyby v kódu během vývoje, což výrazně zvyšuje jeho kvalitu a usnadňuje údržbu. TypeScript také obsahuje spoustu velmi užitečných funkce, jako jsou například dekorátory nebo interfacy, které usnadňují tvorbu robustních a komplexních aplikací.

Díky tomu, že NestJS je postavený nad TypeScriptem, je možné psát přehlednější kód a provádět jednodušší refaktoring, což vede ke kódové bezpečnosti a méně chybám [5], což následné výrazně urychluje vývojový proces a zlepšuje kvalitu kódu.

Kromě toho, TypeScript se snaží být co nejvíce podobný jazyku C#, což může být pro některé vývojáře velkou výhodou, zejména pro ty, kteří jsou zvyklí pracovat s C# a jeho frameworkem pro webové aplikace ASP.NET. Tato podobnost umožňuje snadnější přechod mezi oběma platformami a usnadňuje spolupráci v týmu, kde jsou někteří vývojáři zvyklí na C# a jiní na TypeScript. To je v případě aplikace JISPEN obrovská výhoda, protože její údržbu a případné rozšíření bude provádět společnost Inisoft s.r.o., jejíž programátoři jsou zvyklí pracovat s C# a .NET. To znamená, že s NestJS budou

schopni rychleji a efektivněji vyvíjet a udržovat aplikaci, což může přinést výhody v podobě snížených nákladů a rychlejšího vývoje.

3.2.3 API

Existuje několik způsobů, jak vytvořit a komunikovat s API. Dva nejčastější způsoby jsou REST API a GraphQL. REST API využívá HTTP protokol a pracuje s konkrétními zdroji, zatímco GraphQL umožňuje klientům specifikovat, jaká data chtějí získat, což může vést k menšímu množství dotazů na server.

Výběr mezi REST a GraphQL závisí na konkrétních potřebách a požadavcích aplikace. REST API se často používají pro jednoduché a přímé dotazy na data, kdy jsou výstupní data pevně daná a klient ví, co očekávat. Tento přístup se hodí zejména pro jednoduché webové stránky a mobilní aplikace, které potřebují rychlý a efektivní přístup k datům.

Na druhou stranu, GraphQL je vhodnější pro složitější aplikace, kde klient potřebuje specifikovat, jaká data chce získat a v jakém formátu. To umožňuje klientovi, aby obdržel všechna potřebná data jedním dotazem, což může snížit zátěž na serveru a zlepšit výkon. Nicméně, s použitím GraphQL se často pojí složitější návrh API a větší nároky na správu a údržbu.

NestJS podporuje oba způsoby implementace API a umožňuje vývojářům vybrat si ten, který nejlépe vyhovuje potřebám jejich aplikace. V případě vývoje aplikace JISPEN jsem se rozhodl pro REST API, neboť API aplikace bylo navrženo s ohledem na jednoduchost použití a s ohledem na možnost vývoje dalších klientů, jako například mobilní aplikace. S použitím REST API je možné jednoduše získat data z API pomocí standardních HTTP dotazů, bez nutnosti komplikovanějšího dorozumívání s API serverem.

3.3 UŽIVATELSKÉ ROZHRANÍ

Stejně jako s databází a backendem jsem ponechal React a Next.js jako hlavní technologie pro vývoj uživatelského rozhraní.

3.3.1 React

React je JavaScriptová knihovna pro tvorbu uživatelských rozhraní (UI), která byla vytvořena společnosti Facebook. Tato knihovna je velmi oblíbená mezi vývojáři a používá se k vytváření složitých, dynamických a interaktivních webových aplikací. React umožňuje vytvářet opakovaně použitelné komponenty uživatelského rozhraní a díky tomu je snadné s nimi pracovat

a měnit je podle potřeby, navíc React poskytuje implementaci virtuálního DOM, protože umožňuje efektivnější aktualizaci uživatelského rozhraní bez nutnosti kompletního překreslení celé stránky [6]. Virtualní DOM je abstraktní vrstva mezi reálným DOM a logikou aplikace, která umožňuje Reactu sledovat změny v UI a aktualizovat pouze ty části DOM, které se změnily. Díky těmto vlastnostem je React velmi vhodný pro vytváření moderních a škálovatelných webových aplikací.

React byl vytvořen s cílem usnadnit vývoj webových aplikací v době, kdy se tento proces stává stále složitějším a náročnějším. Tato knihovna umožňuje vývojářům vytvářet uživatelsky přívětivá a výkonná rozhraní rychleji a jednodušeji, než by tomu bylo s tradičním JavaScriptem. Navíc díky využití virtuálního DOMu mohou React aplikace rychleji reagovat na uživatelské vstupy a změny v datových sadách (které docela často bývají velké), což zaručuje vysokou rychlost odezvy aplikace [7].

3.3.2 Next.js

Next.js je framework postavený na React.js, který se specializuje na tvorbu webových aplikací s použitím renderování jednotlivých stránek na straně serveru (SSR) [8]. Tento framework nabízí sadu nástrojů a funkcí, které usnadňují tvorbu vysokovýkonových webových aplikací optimalizovaných pro vyhledávače (SEO) a uživatelskou zkušenost (UX). Díky server-side renderingu, optimalizaci obrázků a další řadě funkcí Next.js zaručí, ze uživatelé vidí úplně načtenou stránku v co nejkratším čase a vyhledávače mají snazší práci s indexací obsahu.

3.3.3 Shrnutí technologií

React a Next.js jsou výkonné nástroje pro tvorbu moderních webových aplikací. Nabízejí řadu výhod oproti jiným nástrojům pro vývoj front-endu, včetně:

- Deklarativní programovací model, který usnadňuje tvorbu znovupoužitelných UI komponent
- SEO-přátelský design
- Implementace virtuálního DOM, která usnadňuje tvorbu rychlých a efektivních webových aplikací
- Výkonná sada nástrojů pro ladění a testování webových aplikací
- Bohatý ekosystém knihoven a nástrojů třetích stran

Jako jednu z dalších výhod lze uvést, že React a Next.js jsou vhodné pro vývojáře různé úrovně zkušeností. Zatímco zkušení vývojáři mohou využívat pokročilé funkce a nástroje k vytváření výkonných a složitých webových aplikací, začátečníci si mohou snadno naučit základy těchto technologií a postupně postoupit ke složitějším projektům. Díky rozsáhlé komunitě, která kolem těchto technologií vyrostla, je také snadné najít řešení problémů, stejně jako podporu a inspiraci pro vytváření nových funkcí a projektů.

3.3.4 Jednoduchost použití

Jednou z nejdůležitějších vlastností aplikace JISPEN je její snadnost použití. Aby byla aplikace co nejefektivnější a co nejvíce uživatelsky přívětivá, bylo při jejím návrhu bráno v úvahu mnoho faktorů.

Každá aplikace, která pracuje s daty, by měla mít validaci zapisovaných údajů. **Validace** zajišťuje, že uživatel zadává jen správná data do polí a tím se minimalizuje riziko vzniku chyb. V aplikaci JISPEN by měla validace dat být zahrnuta do všech formulářů, kde je to nutné, například polí pro emaily, hesla, množství odpadu, pro datum atd. Pokud uživatel zadá nesprávné údaje, aplikace ho upozorní a umožní mu opravu.

Pro zjednodušení práce s aplikací je možné použít číselníky. **Číselníky** obsahují všechny potřebné informace, které se v praxi používají. Uživatel pak nemusí znát a pamatovat si kódy odpadů či nakládaní, stačí jen vybrat odpovídající položku v nabízeném seznamu. To zajišťuje, že uživatel vkládá do aplikace pouze správná data a minimalizuje to riziko vzniku chyb. V rámci aplikace JISPEN jsou použité číselníky PSČ, ZÚJ, částečný katalog odpadů a kódů nakládaní.

Aby uživatel mohl rychle a snadno vyplňovat formuláře, je možné v aplikaci použít našeptávače. **Našeptávač** začne nabízet možnosti na základě zadaných znaků, když uživatel začne do příslušného pole psát. To usnadňuje a zrychluje vyplňování formulářů, protože uživatel nemusí zapisovat celé údaje, ale stačí vybrat odpovídající položku z nabízených možností. Jedním z možných použiti by mohl být našeptavač šablon nebo oprávněných osob při vytvářeni evidence.

Celkově by aplikace JISPEN měla být navržena tak, aby minimalizovala administrativní zátěž. Díky využití validace dat, číselníků a našeptávačů budou uživatelé moci rychle a snadno vyplňovat formuláře s minimálním rizikem vzniku chyb.

3.4 INFRASTRUKTURA

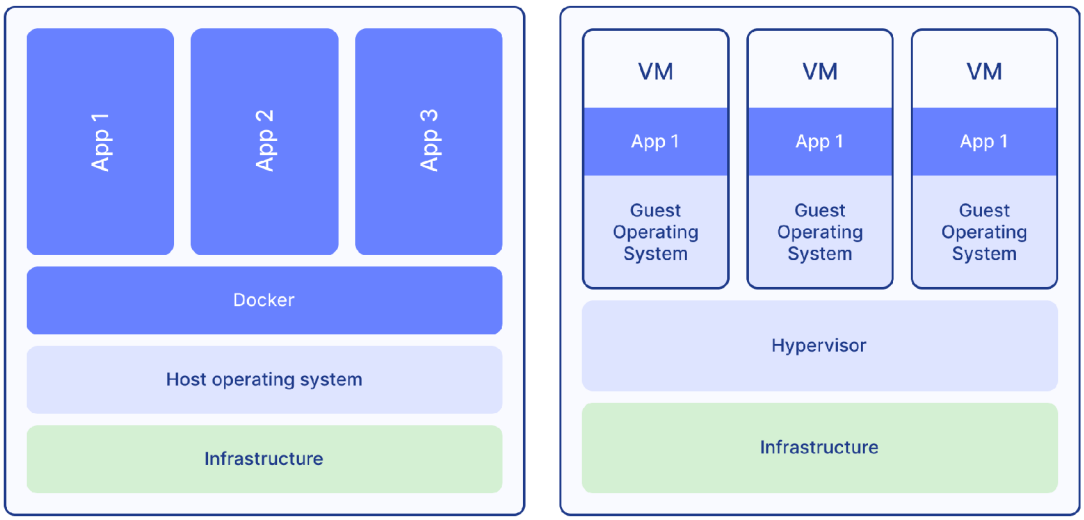
V dnešním rychle se rozvíjejícím světě vývoje softwaru může být obtížné držet krok s neustále se vyvíjejícím technologickým prostředím. Potřeba zajistit kompatibilitu napříč různými zařízeními a platformami je jedním z největších problémů, kterým vývojáři čelí. Existuje mnoho nástrojů pro tyto účely, proto v této kapitole budou uvedeny některé z nich, které jsou vhodné pro aplikaci JISPEN.

3.4.1 Docker. Je správnou volbou?

Docker je silná a inovativní platforma pro vývoj, nasazení a spouštění aplikací v kontejnerizovaném prostředí. S Dockerem je možné snadno zabalit libovolnou aplikaci a všechny její závislosti do jednotného, samostatného celku (dále jen **kontajner**) [9], což umožňuje přenositelnost aplikace na jakýkoliv stroj, který podporuje Docker. Díky tomu Docker umožňuje jednoduše nasazovat aplikace bez ohledu na podkladový hardware nebo software a ušetřit tak čas a náklady na konfiguraci a údržbu infrastruktury.

Hlavni alternativou Dockeru je virtualizace (VM), proti které ale Docker má spoustu výhod:

- Rychlost: Docker spouští aplikace rychleji než VM, protože není nutné emulovat celý operační systém. Docker kontejner sdílí jádro s hostitelem a sdílí také některé další prostředky, což znamená, že je spouštění kontejnerů rychlejší.
- Snadnější správa: Docker je snadněji spravovatelný než VM. Pomocí Dockeru jde snadno vytvářet izolovaná prostředí pro testování a vývoj aplikací, což umožňuje přizpůsobit se různým prostředím a taky se zkracuje doba potřebná k vytvoření a nasazení nových verzí aplikací, což vede k rychlejšímu a efektivnějšímu vývoji.
- Snadné sdílení: Docker **image** (šablona k vytváření kontejnerů) lze snadno sdílet a znovupoužít. To znamená, že lze snadno sdílet celé prostředí aplikace včetně všech závislostí a konfigurace, což usnadňuje spolupráci na vývoji a testování.
- Bezpečnost: Kontejnery Dockeru jsou izolované od hostitelského systému a od jiných kontejnerů, což poskytuje dodatečnou vrstvu zabezpečení pro vaši aplikaci.



Obrázek 3.2: Docker oproti VM

Na závěr lze říci, že Docker je výkonná platforma se spoustou výhod. Aplikace, které jsou kontejnerovány pomocí Dockeru zvyšují produktivitu vývoje a taky přináší škálovatelnost, zabezpečení a mnohem snazší nasazení a správu aplikací napříč platformami a stroji. Docker spravuje aplikace efektivním a spolehlivým způsobem, proto je pro JISPEN ideálním řešením.

3.4.2 Docker Compose

Docker je velmi užitečný pro vytváření izolovaných prostředí pro aplikace a jejich závislosti. Pomocí **Docker Compose** se dá definovat a spustit více kontejnerů najednou a nastavit propojení mezi nimi, díky tomu umožňuje snadné nasazení a správu aplikací v kontejnerech.

Další důležitou funkcí Dockeru je networking. Docker umožňuje vytvářet propojení jak mezi kontejnery a hostem tak i mezi kontejnery. Umožňuje to snadnou komunikaci mezi aplikacemi běžícími v různých kontejnerech, což je pro JISPEN velmi důležité.

Volumy jsou taky významnou funkcí Dockeru, která umožňuje ukládat data mezi hostem a kontejnerem. Díky této funkci jsou data perzistentní i po zastavení a restartu kontejnerů. V aplikaci JISPEN jsou všechna data uložena pomocí Volumů, což zajišťuje rychlé a bezpečné zálohování dat, a to buď při nasazení nových verzí, nebo v případě výpadku.

V Docker Compose lze jednoduše definovat každý kontejner a jeho konfiguraci v jednom souboru YAML. Soubor obsahuje informace o každém

kontejneru, jako jsou jméno kontejneru, použitý obraz, konfigurační proměnné, porty, připojení k externím síťovým rozhraním, vazby mezi kontejnery a další informace. YAML formát je dobrý pro definování kontejnerů v Docker Compose, protože umožňuje jednoduchý a čitelný zápis dat, což zjednodušuje údržbu a aktualizaci souboru a taky snadné definovaní hierarchické struktury dat.

3.4.3 Nginx

Reverzní proxy server je jednou z klíčových součástí infrastruktury aplikace JISPEN. Slouží jako vstupní bod pro aplikační servery a zajišťuje jejich bezpečnost a ochranu. Existuje několik řešení, z nichž jedním z nejlepších je Nginx - špičkový reverzní proxy server s vysokou rychlostí, flexibilitou a širokou škálou funkcí, včetně podpory HTTPS a HTTP/2.

Jedním z hlavních důvodů, proč Nginx byl vybrán jako entrypoint do aplikace, je zajištění bezpečnosti. Nginx dokáže filtrovat a blokovat nebezpečné požadavky, chránit aplikaci před útoky a bránit neoprávněnému přístupu. Díky tomu můžeme mít jistotu, že aplikace je chráněna a že data na ní uložená jsou v bezpečí.

Dalším důvodem, proč je Nginx vhodným řešením, je jeho schopnost fungovat jako load balancer, který umožňuje rovnoměrné rozdělení zátěže mezi různé servery, což zvyšuje dostupnost a výkon aplikace. Aplikace JISPEN sice není příliš zatížená, nicméně je dobré mít připravené řešení, které lze v budoucnu snadno použít.

3.5 EXPORT

3.5.1 XML pro ostatní aplikace

Realizaci exportu lze efektivně řešit pomocí jazyka XML, protože je výkonný a flexibilní pro ukládání a přenos dat, používá se pro různé aplikace a platformy (včetně programů ENVITA a EVI 8).

Výhodou XML pro programátora je, že jeho syntaxe je jednoduchá a srozumitelná, což umožňuje snadno číst a upravovat data. Navíc tento formát umožňuje vytvářet vlastní značky (tagy) [10], které popisují data a jejich strukturu a také má standardizovaný způsob validace, což znamená, že data mohou být snadno přenášena a zpracovávána.

Formát XSD (XML Schema Definition) byl použity pro validaci XML souborů. Použití XSD pomáhá automatizovat procesy a odstraňovat lidské

chyby při kontrole dat, čímž se zvyšuje efektivita a spolehlivost procesů, které pracují s daty v XML [11]. Vzhledem k tomu, že exportované soubory budou primárně používány s programy společnosti Inisoft s.r.o. (ENVITA a EVI 8), byl pro správný formát XSD použit XSD soubor této společnosti, aby se zajistilo, že data v XML budou správně strukturována a formátována, díky čemu se zajišťuje jednotnost datové struktury, což usnadní přenos dat a minimalizuje chybovost.

Toto řešení zajistí jednotnost dat, což je klíčové pro zajištění správného přenosu informací mezi různými systémy. Zároveň umožní uživatelům jednoduché a přehledné zpracování dat a minimalizuje možnost chyb při přenosu informací.

Celkově lze tedy říci, že kvalitní rozhraní pro přenos vytvořených dat je důležitou součástí aplikace JISPEN a umožňuje uživatelům snadnou a efektivní práci s informacemi o odpadech.

3.5.2 XLSX pro uživatele

Exportování souborů pro různé systémy není jediným využitím této funkcionality v aplikaci JISPEN. Uživatelé této aplikace by měli mít taky možnost exportovat svá data do souboru pro vlastní potřebu nebo k uložení a použití offline.

Moderním formátem pro správu a prezentaci dat je formát XLSX. Je nástupcem staršího formátu XLS a vyznačuje se vysokou mírou kompatibility s různými aplikacemi a operačními systémy [12]. Formát XLSX podporuje širokou škálu funkcí, včetně formátování buněk, vzorců, grafů a kontingenčních tabulek, což uživatelům umožňuje vytvářet interaktivní prezentace dat, které jsou pro publikum poutavé.

Tento formát je jedním z nejoblíbenějších formátů pro správu a prezentaci dat, a proto byl zvolen jako další exportní formát JISPEN, pro komplexní a uživatelsky přívětivý přehled dat.

4 REALIZACE ŘEŠENI

Realizace řešení je klíčovou fází vývoje softwarového projektu, která spojuje teorii s praxí. V této fázi jsou implementovány nápady a plány z předchozích fází vývoje do skutečného produktu. V této kapitole se zaměříme na praktickou část vývoje aplikace JISPEN, která zahrnuje implementaci navržených funkcí a testování vytvořených řešení. Budeme se věnovat použitým technologiím, konfiguraci infrastruktury a samotné implementaci.

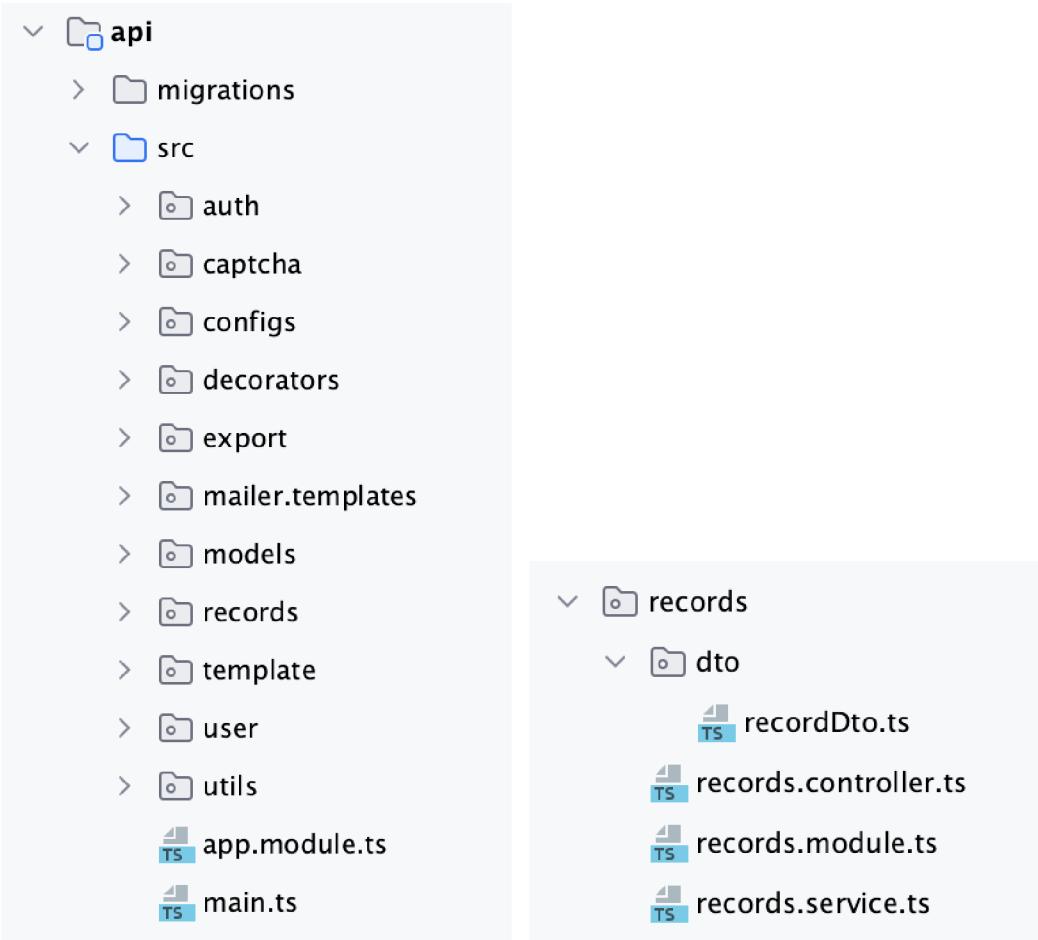
4.1 SERVEROVÁ ČÁST

4.1.1 Struktura

Architektura aplikace JISPEN byla navržena tak, aby byla co nejvíce modulární a snadno rozšiřitelná. V celém programu jsou jednotlivé funkcionality rozděleny do samostatných domén, které zahrnují:

- Přihlášení a registrace
- Šablony
- Evidence
- Export

Každá doména má vlastní kontrolér, servisu, modul a DTO. Kontroléry se používají k přijímání požadavků od uživatelů a jejich směrování na příslušnou službu, která pak požadavek zpracuje a vrátí výsledek. Modul se používá pro registraci domény a definuje vazby mezi různými doménovými prvky a entitami. DTO popisuje strukturu vstupních dat zasílaných do kontroléru. Díky této struktuře kódu je snadné provádět údržbu, rozšiřovat aplikaci o nové funkcionality a zlepšovat tak celkovou modularitu a přehlednost aplikace JISPEN.



Obrázek 4.1: Struktura backend části

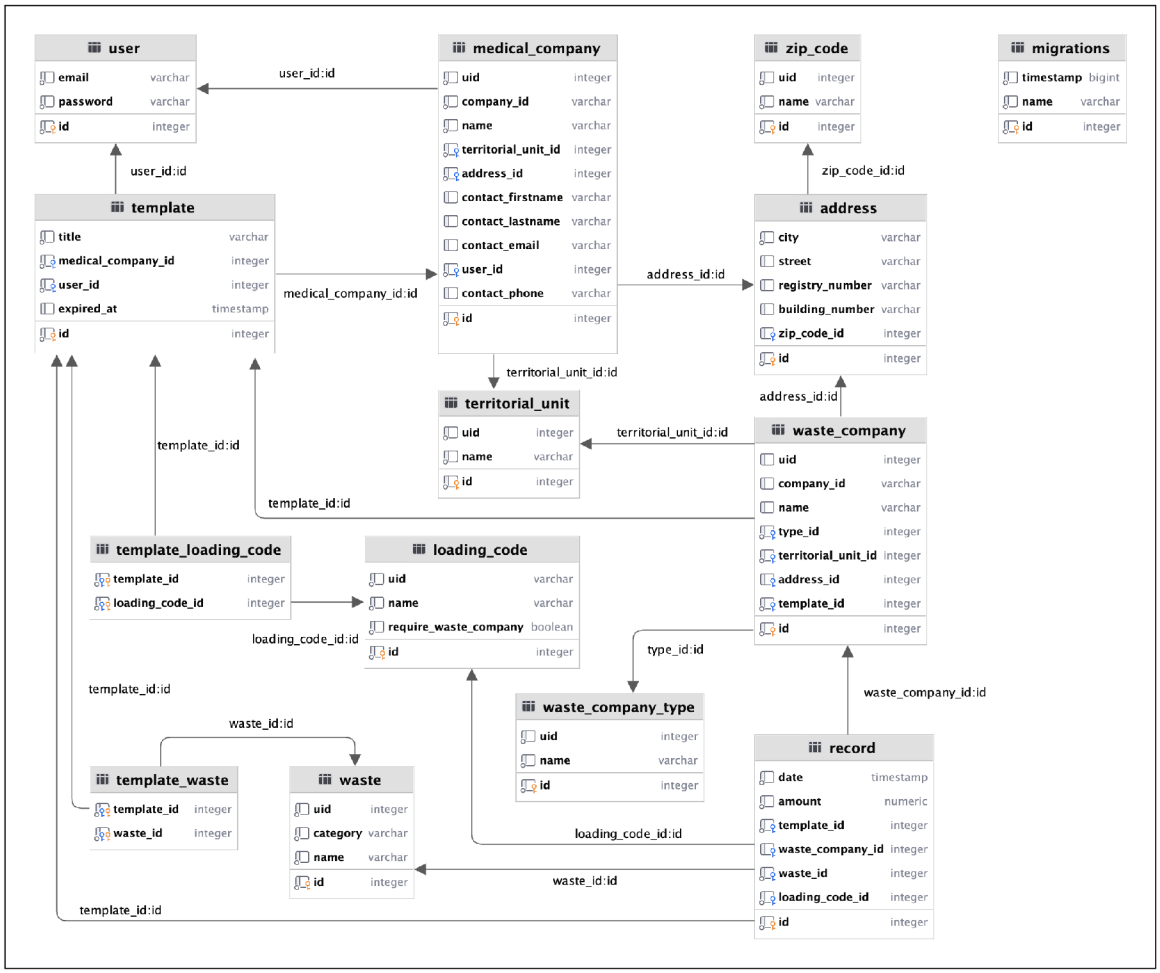
4.1.2 Export

Během vývoje jsem se nesoustředil pouze na zlepšení existující funkcionality MVP, ale také jsem se musel vypořádat s novým úkolem – vytvořit exportní funkcionalitu. Původně bylo potřeba exportovat data pouze ve formátu XML, který byl následně nahráván do systému ENVITA nebo EVI 8. Během manuálního testování s několika zákazníky se však objevily požadavky na rozšíření funkcionality exportu, takže jsem musel vytvořit další typ exportu, tentokrát ve formátu XLSX (Excel Spreadsheet neboli Excelová tabulka), který uživatelům poskytuje rychlý přehled o všech datech na jejich účtu.

Pro generování exportních souborů byly použity knihovny **xmlbuilder2** a **node-xlsx**, které umožňují snadnou a bezpečnou tvorbu výstupních souborů dle podmínek zákazníka. Díky těmto knihovnám mohu snadno a rychle generovat XML a XLSX soubory s přesně definovanou strukturou dat a zajistit tak maximální přesnost a spolehlivost exportní funkcionality.

4.1.3 Databáze

Během vývoje aplikace a upřesňování požadavků zákazníka došlo ke změně původního schématu databáze, která byla součástí MVP aplikace, vyvinuté během ročníkového projektu. Finální verze schématu databáze je proto navržena tak, aby byla snadno udržovatelná a případně rozšiřitelná v budoucnu, a aby plně odpovídala potřebám zákazníka. Díky této přístupnosti může být aplikace snadno přizpůsobena budoucím požadavkům a rozšíření funkcí, což zvyšuje její efektivitu a využitelnost pro uživatele.



Obrázek 4.2: Finální struktura databáze

4.1.4 Zabezpečení

Bezpečnost je klíčovým faktorem každého backendu a proto se v této kapitole podrobně věnuji tomu, jak byla zajištěna bezpečnost v rámci aplikace JISPEN.

Jedním z hlavních prvků zabezpečení je autentizace. Pro tyto účely bylo zvoleno ověřování pomocí JWT (JSON Web Token) tokenů. Pro ochranu

různých API endpointů je implementován speciální guard – JwtAuthGuard. Tento guard slouží k ověření platnosti JWT tokenu, který je přenášen v hlavičce HTTP požadavku. Pokud je token platný a autorizuje uživatele k přístupu k danému zdroji, guard umožní přístup k danému API endpointu. Pokud token neplatí nebo neumožňuje přístup k danému zdroji, guard vrátí chybovou odpověď. Pro práci s JWT tokeny je v NestJS používán modul Passport.js, který umožňuje vytvářet různé strategie pro autentizaci. Pro ověření platnosti JWT tokenu v aplikaci JISPEN je využívána strategie Jwt-Strategy, která využívá algoritmu HS256 pro dekódování a ověření podpisu tokenu.

Pro zabezpečení dat uživatelů jsou v aplikaci používány další techniky, jako například hashování hesel pomocí algoritmu bcrypt. Tento algoritmus je speciálně navržen pro ukládání hesel v bezpečné podobě a umožňuje ukládat hesla v databázi v šifrované podobě. Díky tomu minimalizuje riziko úniku hesel v případě prolomení databázového zabezpečení nebo pokusu o neoprávněný přístup. Pro ověření hesla uživatele při přihlašování je používána stejná funkce hashování, která porovnává vstupní heslo s uloženým hashem v databázi. Tím je zajištěno, že heslo je správně ověřeno, aniž by bylo nutné uchovávat samotné heslo v otevřené podobě. Tato bezpečnostní opatření jsou důležitá pro ochranu citlivých dat uživatelů a zajišťují, že data jsou uchovávána v bezpečí před potenciálními hrozbami.

Další důležitou součástí zabezpečení aplikace je validace dat, která přicházejí z uživatelské častí. Aplikace používá různé techniky pro validaci vstupů, například kontrolu délky a formátu řetězců nebo kontrolu správnosti e-mailových adres. Tím se minimalizuje riziko útoků, jako jsou SQL injection nebo cross-site scripting (XSS).

Kromě toho je aplikace chráněna proti SQL útokům díky používání ORM TypeORM, což umožňuje vytváření dotazů na databázi pomocí objektů a metod namísto ručního psaní SQL dotazů. To minimalizuje riziko chyb v syntaxi SQL dotazů a zabraňuje útokům, které by mohly zneužít chyby v syntaxi SQL dotazů k ovládnutí databáze.

V neposlední řadě byla aplikace průběžně aktualizována během vývoje s cílem minimalizovat riziko výskytu známých bezpečnostních zranitelností a hrozeb. Tyto aktualizace zahrnovaly jak aktualizace softwarových balíků a knihoven, tak i opravy chyb a zranitelností nalezených během testů. Díky pravidelným aktualizacím je aplikace schopna přinášet vysokou úroveň zabezpečení a spolehlivosti pro uživatele.

4.2 CAPTCHA

Captcha je bezpečnostní prvek webových formulářů, který pomáhá oddělit uživatele od automatických botů. Úkolem Captcha je ověřit, že vstupní formulář vyplňuje skutečný člověk a ne počítačový program, který by mohl například zneužít systém nebo spamovat. Captcha má několik druhů, včetně optických (například rozpoznávání čísel nebo obrázků), akustických nebo matematických. Nejpopulárnější je však captcha založená na rozpoznávání textu.

Existuje mnoho open-source i komerčních řešení pro captcha, jako je Google reCAPTCHA nebo hCaptcha, které jsou běžně používány na webových stránkách po celém světě. Tato řešení jsou obecně velmi efektivní, ale mají řadu nedostatků, které mohou být pro některé použití kritické. V případě aplikace JISPEN bylo rozhodnuto, že nebude použito žádné existující řešení captcha, protože tyto služby jsou provozovány na externích serverech a v případě použití aplikace JISPEN v intranetech by bylo nutné mít přístup k internetu, což by mohlo být problematické.

Namísto toho bylo rozhodnuto vytvořit vlastní captcha. Při implementaci vlastního řešení byl zvolen Redis, což je rychlé a účinné úložiště klíč-hodnota, které se často používá jako cache nebo pro řešení podobných úloh. Kvůli jeho rychlosti a možnosti nastavení TTL (time-to-live), což umožňuje automatické mazání starých klíčů po určité době, přispívá k úspornosti paměti a zvyšuje bezpečnost. Redis je také velmi škálovatelný a snadno přizpůsobitelný, což umožňuje vytvořit řešení, které je obzvláště vhodné pro požadavky aplikace JISPEN.

Vlastní captcha aplikace JISPEN je úmyslně navržena jednoduše, ale tak, aby dostačovala svému účelu. Její text se skládá z náhodně generovaných číslic, délka se pohybuje mezi 4 a 8 číslicemi. Po vygenerovaní se text ukládá do Redisu a následné se zobrazuje uživateli, který musí přepsat tento text do textového pole na webové stránce, aby mohl dokázat, že je skutečným uživatelem a má oprávnění aplikaci používat. Pokud uživatel zadá nesprávný text, aplikace mu vygeneruje nový kód, smaže starý a umožní zadat znovu.

Implementace vlastního řešení byla vhodnou volbou, protože tato aplikace může být používána v rámci uzavřených firemních sítí a intranetů, kde není vhodné používat podobné online nástroje jako Google reCAPTCHA, kvůli potenciálním bezpečnostním rizikům a omezení přístupu k těmto službám. Další výhodou je plná kontrola nad algoritmem a jeho přizpůsobení specifickým potřebám aplikace. Navíc, vlastní řešení je plně integrováno do aplikace a nemusí se spoléhat na externí služby, což zvyšuje bezpečnost a snižuje závislost.

4.3 WEBOVÉ ROZHRANÍ

V této kapitole se budeme věnovat popisu frontendové části aplikace JISPEN. Konkrétně se budeme zabývat technologiemi, které byly využity při vývoji, použitými knihovnami a nástroji pro zabezpečení a správu uživatelských dat.

4.3.1 Technologie a knihovny

Jak bylo uvedeno v kapitole 4.3, pro vývoj frontendu byl použit framework **React** v kombinaci s **Next.js**, což umožňuje snadnou tvorbu server-side renderovaných aplikací. Díky tomuto přístupu se zajišťuje rychlost načítání a optimalizace SEO.

Pro tvorbu UI byli použité knihovny jako **Material-UI**, která poskytuje mnoho předpřipravených UI komponent a umožňuje snadnou tvorbu responzivního designu a **TailwindCSS**, která poskytuje připravené CSS třídy pro různé styly a umožňuje tak snadnou tvorbu vlastních komponent. Při tvorbě uživatelského rozhraní byl kladen důraz na responzivitu a jednoduchost ovládání. Všechny stránky byly navrženy tak, aby byly snadno přehledné a uživatelsky přívětivé. Díky použití Material-UI a TailwindCSS bylo dosaženo moderního vzhledu a rychlého vývoje. V rámci návrhu uživatelského rozhraní byla vytvořena podrobná specifikace layoutu pro každou stránku, včetně umístění a stylizace jednotlivých komponent. Byla také vytvořena sada společných prvků a komponent pro jednotný vzhled aplikace. Celkově bylo cílem vytvořit uživatelsky přívětivé a moderní rozhraní, které bude snadno ovladatelné pro uživatele a zároveň bude odpovídat požadavkům projektu.

Pro efektivní zpracování dat, získaných z API, a minimalizaci množství dotazů na server existuje mnoho různých nástrojů a knihoven. Jednou z těchto knihoven je **SWR**, která je speciálně navržena pro práci s daty z API. SWR je knihovna pro React, která umožňuje snadné získávání a aktualizaci dat ze serveru v reálném čase. Jedním z hlavních přínosů této knihovny je minimalizace množství dotazů na server díky cachování dat. Pokud je například potřeba zobrazit seznam všech šablon uživatele v aplikaci, tak se při prvním načtení stránky získají data z API a uloží se do paměti SWR pro další použití. Pokud uživatel přejde na jinou stránku a následně se vrátí zpět na seznam šablon, data budou již načtena z paměti a nebude nutné provádět nový dotaz na server. Další přínos SWR je jeho schopnost pracovat s asynchronními funkcemi a poskytnout různé možnosti pro řízení chování při získávání dat, jako například nastavení intervalu pro periodické načítání dat. V rámci aplikace JISPEN SWR umožnilo snadnou manipulaci s daty, což bylo užitečné při tvorbě formulářů.

V aplikaci byly dále využity různé nástroje pro správu stavu a autentizaci uživatelů. K ukládání uživatelského stavu byly použity cookies a také knihovna **jotai**, která umožňuje snadnou správu a aktualizaci stavu aplikace pomocí kontextu. Díky tomuto přístupu bylo možné ukládat informace o uživateli a jeho nastavení při práci s formuláři a zároveň snadno aktualizovat stav aplikace bez nutnosti komunikace se serverem. Pro autentizaci uživatelů byl využit JWT token, který sloužil k ověření identity uživatele při každém dotazu na server. Po přihlášení uživatele byl na serveru vygenerován token obsahující informace o uživateli a jeho právech. Tento token byl poté ukládán do cookies a automaticky přidáván ke každému dotazu na server. Díky tomuto přístupu byl zajištěn bezpečný a spolehlivý způsob autentizace uživatelů v aplikaci.

4.3.2 Mobilní aplikace

V rámci vývoje uživatelského rozhraní pro prací s aplikaci JISPEN byla implementována funkčnost pro podporu progresivních webových aplikací. **PWA** je moderní technologie, která umožňuje uživatelům přistupovat k webovým aplikacím jako ke standardním mobilním aplikacím.

Mezi hlavní výhody PWA patří možnost použití aplikace, aniž by uživatel byl připojen k internetu, rychlost odezvy díky možnosti cachování dat a jednoduchá instalace přímo z webové stránky na plochu nebo domovskou obrazovku zařízení. I když PWA nabízí mnoho výhod, v JISPEN nebylo možné všechny využít. Například pro práci s API musí být uživatel stále online, což může být omezující faktor pro některé uživatele. Nicméně, hlavním důvodem pro použití PWA v JISPEN je usnadnění přístupu k aplikaci a zlepšení uživatelské zkušenosti. Díky PWA uživatel nemusí otevírat prohlížeč a zadávat adresu, což ušetří čas a zjednoduší práci s aplikací.

PWA však také má několik nevýhod, jako je menší podpora některých funkcí, například notifikace, a to vzhledem k tomu, že PWA není zcela samostatnou aplikací. Nicméně PWA lze používat na různých zařízeních a v různých webových prohlížečích. Podporu PWA má například Google Chrome, Mozilla Firefox, Opera nebo Safari. Na mobilních zařízeních lze PWA instalovat jako aplikaci přímo z prohlížeče.

V rámci vývoje webového rozhraní pro aplikaci JISPEN byla technologie PWA implementována díky možnostem poskytovaným frameworkem Next.js. Pomocí knihovny **next-pwa** bylo možné jednoduše přidat podporu PWA do aplikace JISPEN a tím zlepšit uživatelskou zkušenost při používání aplikace.

4.4 INFRASTRUKTURA

V této kapitole se zaměříme na technické aspekty aplikace JISPEN v rámci její infrastruktury. Konkrétně se podíváme na praktické použití Dockeru, Nginx a verzování kódu. Tyto nástroje byly klíčové pro úspěšné nasazení aplikace a její bezproblémový chod v produkčním prostředí.

4.4.1 Praktické použiti Dockeru

Aplikace JISPEN využívá obě zmíněné technologie, díky nimž bylo možné vytvořit několik prostředí pro vývoj, testování a produkci. Konkrétně se jedná o následující prostředí:

- Vývojové (lokální) prostředí - určené pro vývoj aplikace
- Testovací (develop) prostředí - používané pro manuální testování aplikace před nasazením na produkční server
- Produkční (production) prostředí - hotová verze aplikace, kterou používají uživatelé

Díky Dockeru jsou všechna tři prostředí snadno přenositelná mezi různými operačními systémy. Například vývoj aplikace probíhal na MacOS, zatímco produkční a testovací servery mají Linux Debian, nicméně aplikace se taky spouštěla na operačním systému Windows pro lokální testování.

Jednou z výhod Docker Compose je možnost sdílet obsah různých konfiguračních souborů mezi sebou, což pomáhá předejít duplicitě dat v konfiguračních souborech. Při vývoji aplikace JISPEN byly použity tři soubory konfigurací pro Docker Compose:

- **docker-compose.yml** - základní konfigurační soubor, obsahuje základní konfigurace, které jsou sdíleny mezi ostatními konfiguračními soubory. Tento soubor umožňuje definovat jednotlivé služby a jejich závislosti, ale také konfigurace sítě, volitelné argumenty a další parametry pro spuštění aplikace.
- **docker-compose.override.yml** - používá se pro upřesňování základní konfigurace pro vývojové prostředí. Docker Compose ho automaticky použije při běhu standardního příkazu **docker-compose up**.
- **docker-compose.prod.yml** - používá se pro produkční a develop prostředí. Tento soubor upravuje konfigurace, které jsou specifické pro produkční prostředí, jako jsou například menší Docker Images a optimalizace kódu.

```yaml
version: '3.8'
services:
    db:
        image: postgres:14
        container_name: jispen-postgres
        env_file:
            - ./database/.env
        volumes:
            - db:/var/lib/postgresql/data

    api:
        container_name: jispen-api
        build:
            context: ./api
        depends_on:
            - db
        links:
            - db

    web:
        container_name: jispen-web
        build:
            context: ./web
        depends_on:
            - api
        links:
            - api
```

```yaml
    nginx:
        image: nginx:alpine
        container_name: jispen-nginx
        volumes:
            - ./nginx/default.conf:/etc/nginx/conf.d/default.conf
        ports:
            - "8008:80"
        depends_on:
            - api
            - web
        restart: on-failure

    redis:
        image: redis:7-alpine
        container_name: jispen-redis

volumes:
    db:
```

Obrázek 4.3: Základní konfigurační soubor

4.4.2 Nginx

V Docker Compose je Nginx používán jako další kontejner, který má jeden vnější otevřený TCP port, přes který probíhá komunikace s celou aplikaci JISPEN. Díky tomu se dá zabezpečit přístup k aplikaci pouze skrze Nginx a blokovat ostatní přístupy na síťové úrovni. Tento postup je velmi důležitý pro zabezpečení naší aplikace a minimalizaci možností útoků.

4.4.3 Verzování kódu

Verzování kódu je nezbytnou součástí jakéhokoli vývojového cyklu a umožňuje efektivní řízení změn a integraci nových funkcí do aplikace. V případě aplikace JISPEN byl celkový vývoj prováděn pomocí verzovacího systému Git a podle metodologie Git Flow, která je velmi užitečná pro agilní vývoj. Tato metodologie rozděluje vývojový cyklus na několik fází, což usnadňuje řízení změn a integraci nových funkcí do aplikace.

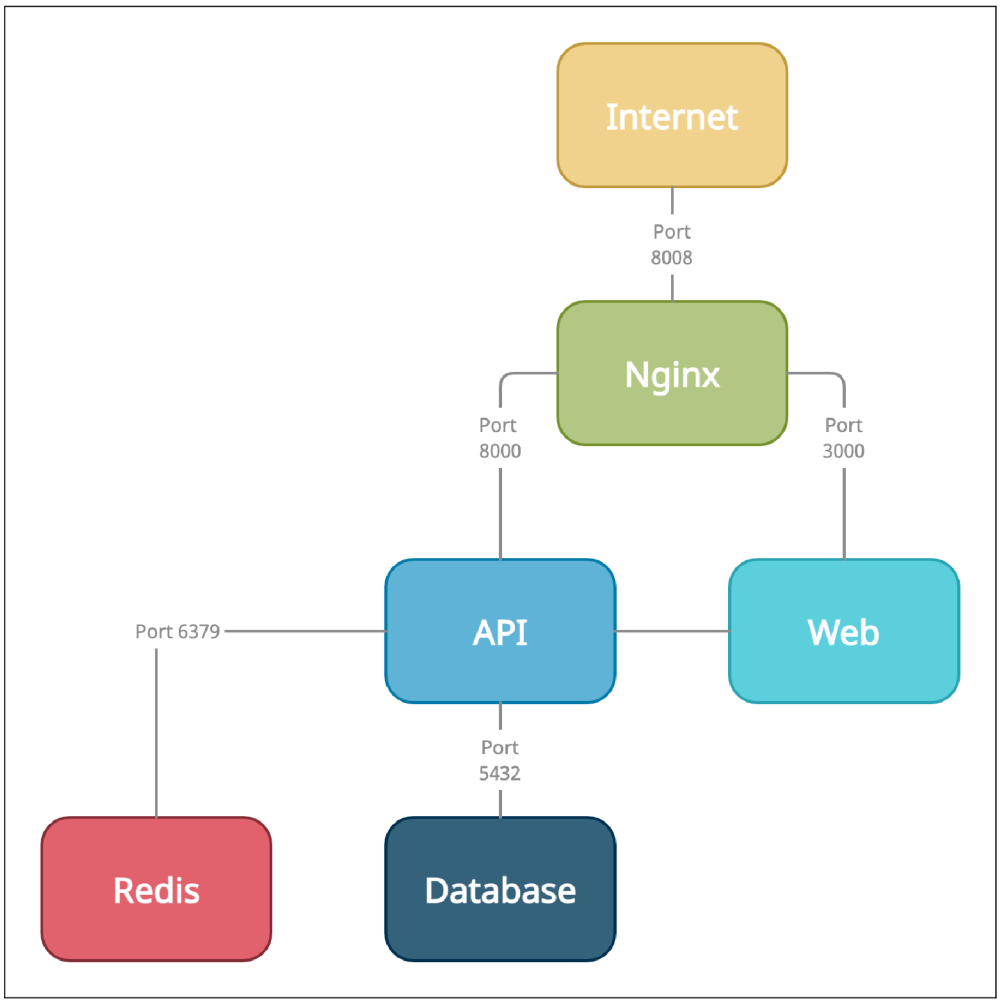
Díky Git Flow bylo možné vytvořit a udržovat různá prostředí pro vývoj, testování a produkci aplikace. Každá větev měla své specifické prostředí, což umožňovalo testování a úpravy větví bez ohrožení vývojových procesů. Tato metodologie také umožňuje snadné řešení chyb v aplikaci a zlepšení celkové kvality kódu.

Metodologie Git Flow se ukázala jako klíčový faktor pro bezpečný vývoj aplikace JISPEN. Díky této metodologii bylo možné řídit vývojové cykly efektivně a zároveň minimalizovat riziko vývojových chyb. Práce probíhala na různých větvích a problémy se řešily samostatně, aniž by ovlivnily ostatní vývojové procesy. To vše umožnilo vytvoření stabilní a bezpečné aplikace, která splňuje očekávání zákazníků.

4.4.4 Architektura a propojení jednotlivých části aplikace

Celkový návrh architektury se v praxi ukázal jako vhodný, bezpečný a dobře rozšiřitelný. Jednotlivé komponenty aplikace byly navrženy tak, aby byly co nejvíce odděleny a zároveň propojeny pomocí standardních rozhraní. Tento přístup umožňuje jednoduchou údržbu a rozšiřování aplikace v budoucnu.

Na následujícím obrázku je zobrazena architektura aplikace, která ukazuje propojení jednotlivých komponent:



Obrázek 4.4: Architektura aplikace

4.5 TESTOVANÍ

4.5.1 Automatizované testování

Po dohodě se zákazníkem nebylo nutné provádět žádné automatizované testy (např. unit nebo e2e) z důvodu nedostatku času ani finančního rozpočtu. Všechny testy byly provedeny konzultantem panem Grusmanem a společností Inisoft s.r.o. Tyto testy byly provedeny s cílem zajistit správnou funkčnost aplikace a zajistit, že všechny požadavky zákazníka jsou splněny.

4.5.2 Prostředí pro ruční testování

Vývojové prostředí bylo nastaveno s třemi hlavními částmi: lokální, vývojové a produkční.

Lokální prostředí slouží pro lokální testování a ladění kódu, zatímco vývojové prostředí bylo založeno na GitLab Actions, kde bylo použito automatické nasazení pomocí Docker Swarm. Docker Swarm poskytuje jednoduchý a bezpečný způsob nasazení aplikací. Navíc umožňuje snadnou správu několika hostitelských strojů a aplikací v nich běžících [13].

Na druhé straně, produkční prostředí bylo spouštěno ručně pomocí Docker Compose, v důsledku požadavku zákazníka. Docker Compose je nástroj pro správu aplikací, které jsou složené z více služeb. Umožňuje jednoduše definovat a spravovat aplikace v dockerových kontejnerech, což umožňuje vyvinout aplikaci na jednom stroji a pak jednoduše nasadit stejnou aplikaci na jiný stroj [13].

Příkaz pro spuštění aplikace je `docker-compose up`. V případě produkčního prostředí je třeba přidat cestu do produkční konfigurace.
Výše uvedený příkaz se používá ke spuštění kontejnerů definovaných v souborech **docker-compose.yml** a **docker-compose.prod.yml** (produkční prostředí). Tento způsob spuštění umožňuje spustit aplikaci s minimálními úpravami a s vysokou mírou důvěryhodnosti.

4.6 MOŽNOSTI DALŠÍHO VYUŽITÍ A ROZŠÍŘENÍ APLIKACE

Po dokončení vývoje a oficiálním spuštěni aplikace JISPEN bylo identifikováno několik možností dalšího využití a rozšíření. Tyto možnosti se týkají především ještě většího zlepšení uživatelského rozhraní, rozšíření funkcionalit aplikace a přidání nových modulů pro správu odpadů.

Jedním z příkladů možností rozšíření aplikace by mohla být implementace modulu pro notifikaci uživatelů o nutnosti provést evidenci odpadů. Tento modul by mohl využívat například e-mailové notifikace, které by uživatele upozorňovaly na blížící se termín pro provedení zápisu do evidence odpadů. Uživatelé by mohli také získat přehled o plánovaných akcích v rámci evidence odpadů a upozornění na případné změny v plánech.

Dalším možným rozšířením aplikace JISPEN by mohlo být vytvoření mobilní aplikace pro uživatele, kteří se většinu času pohybují mimo kancelářské prostory a nejsou on-line. Mobilní aplikace by umožnila uživatelům snadněji a rychleji zadávat informace o produkci odpadů přímo do mobilního zařízení, což by dále zrychlilo celý proces vedení evidence odpadů. Kromě toho by bylo možné využít mobilní notifikace pro upozornění uživatele na nutnost provést evidenci odpadů. Mobilní aplikace by přinesla uživatelům větší flexibilitu a pohodlí při vedení evidence odpadů a zlepšila celkovou efektivitu procesu.

Aplikace by také mohla být rozšířena prostřednictvím implementace dalších modulů pro přehlednější zobrazení statistik, například pro analýzu produkce odpadů v různých obdobích nebo pro sledování aktuálního stavu plnění jednotlivých nádob na odpad. Tyto moduly by umožnily uživatelům získat komplexní informace o vývoji produkce odpadů a plnění nádob a následně podle toho plánovat další kroky v rámci odpadového hospodářství.

Celkově lze tedy říci, že aplikace JISPEN má vysoký potenciál pro další rozvoj a rozšíření o další užitečné funkce a moduly, které by mohly ještě více usnadnit vedení evidence odpadů a snížit administrativní zátěž povinných subjektů.

ZÁVĚR

V této bakalářské práci byl prezentován vývoj aplikace JISPEN, která slouží k průběžnému evidování odpadů ve zdravotnických zařízeních. Aplikace vznikla na základě povinnosti vedení evidence odpadů, která vyplývá z aktuálně platného zákona o odpadech.

Rozpracování a rozšíření teto aplikace mělo za cíl ještě více usnadnit a zefektivnit proces vedení evidence odpadů a tím pomoct snižovat administrativní zátěž subjektům povinných ji vést. Mezi klíčové funkcionality, které byly přidány, patří například export dat do různých formátů s omezením rozsahu za období nebo vybrané evidence, dále rozšíření možností pro editaci nových šablon a úpravu záznamů v již vytvořené evidenci. Uživatelské rozhraní bylo optimalizováno pro různé typy zařízení a také byly zlepšeny mechanismy pro rychlejší a efektivnější práci s aplikací.

Během vývoje aplikace byl kladen důraz na zajištění bezpečnosti dat a dodržení legislativních požadavků v oblasti odpadového hospodářství. Byly použity nejmodernější bezpečnostní technologie a data jsou ukládána v souladu s platnou legislativou.

Výsledkem této práce je plně funkční a uživatelsky přívětivá aplikace JISPEN, která usnadňuje a zefektivňuje proces vedení evidence odpadů pro nemocnice a zdravotnická zařízení. Aplikace byla úspěšně otestována a získala pozitivní ohlas u uživatelů a zákazníka.

LITERATURA

[1] ČESKO. Zákon č. 541/2020 Sb., o odpadecsh, 2020. URL: https://www.zakonyprolidi.cz/cs/2020-541.

[2] Intranet. Intranet — Wikipedia, The Free Encyclopedia, 2008. URL: https://en.wikipedia.org/wiki/Intranet.

[3] Leo Hsu Regina Obe. *PostgreSQL: Up and Running*. O'Reilly Media, Inc., 2017.

[4] Mike Cantelon Alex R. Young, Bradley Meck. *Node.js in Action 2nd Edition*. Manning Publications, 2017.

[5] Boris Cherny. *Programming TypeScript*. O'Reilly Media, Inc., 2019.

[6] Kirupa Chinnathambi. *Learning React: A Hands-On Guide to Building Web Applications Using React and Redux*. Addison-Wesley Professional, 2018.

[7] Meta Platforms, Inc. Oficiální dokumentace, 2023. URL: https://react.dev/learn.

[8] Vercel Inc. Oficiální dokumentace, 2023. URL: https://nextjs.org/docs.

[9] Jeff Nickoloff. *Docker in Action*. O'Reilly Media, Inc., 2016.

[10] Elliotte Rusty Harold and W. Scott Means. *XML in a Nutshell*. O'Reilly Media, Inc., 2003.

[11] Eric van der Vlist. *XML Schema*. O'Reilly Media, Inc., 2001.

[12] Excel 2016 Bible. *XML Schema*. Microsoft Corporation, 2015.

[13] Docker Inc. Oficiální dokumentace, 2013 – 2023. URL: https://docs.docker.com/.

A PŘÍLOHY

A.1 UKÁZKA EXPORTU

	A	B	C	D	E	F	G	H	I	J
1	Datum	Množství	Kód odpadu	Kategorie odpadu	Název odpadu	Osvědčení	Kód nakládání	Název kódu nakládání	Název šablony	Název provozovny
2	15.01.2023	0.002	150101	O	Papírové a lepenkové obaly		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	první	INISOFT s.r.o.
3	20.01.2023	0.001	150101	O	Papírové a lepenkové obaly		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Druhá	Inisoft s.r.o.
4	02.02.2023	0.002	180103	N	Odpady, na jejichž sběr a odstraňování jsou kladeny zvláštní požadavky s ohledem na prevenci infekce		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
5	02.02.2023	0.002	180103	N	Odpady, na jejichž sběr a odstraňování jsou kladeny zvláštní požadavky s ohledem na prevenci infekce		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
6	03.02.2023	0.0005	180101	O	Ostré předměty (kromě čísla 18 01 03)		BN30	Převzetí odpadu od fyzických nepodnikajících osob – občanů	Infekční	INISOFT s.r.o.
7	03.02.2023	0.0005	180101	O	Ostré předměty (kromě čísla 18 01 03)		BN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
8	04.02.2023	0.0001	180104	O	Odpady, na jejichž sběr a odstraňování nejsou kladeny zvláštní požadavky s ohledem na prevenci infekce		B00	Odpad převzatý od původce, jiné oprávněné osoby, nebo jiné provozovny	Infekční	INISOFT s.r.o.
9	04.02.2023	0.0001	180104	O	Odpady, na jejichž sběr a odstraňování nejsou kladeny zvláštní požadavky s ohledem na prevenci infekce		BN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
10	01.01.2023	0.0005	180103	N	Odpady, na jejichž sběr a odstraňování jsou kladeny zvláštní požadavky s ohledem na prevenci infekce		C00	Množství odpadu převedené z minulého roku	Infekční	INISOFT s.r.o.
11	05.01.2023	0.0005	180103	N	Odpady, na jejichž sběr a odstraňování jsou kladeny zvláštní požadavky s ohledem na prevenci infekce		CN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
12	15.01.2023	0.002	180103	N	Odpady, na jejichž sběr a odstraňování jsou kladeny zvláštní požadavky s ohledem na prevenci infekce		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
13	15.01.2023	0.002	180103	N	Odpady, na jejichž sběr a odstraňování jsou kladeny zvláštní požadavky s ohledem na prevenci infekce		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
14	31.01.2023	0.08	200301	O	Směsný komunální odpad		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
15	31.01.2023	0.08	200301	O	Směsný komunální odpad		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
16	31.01.2023	0.05	200101	O	Papír a lepenka		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
17	31.01.2023	0.05	200101	O	Papír a lepenka		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
18	31.01.2023	0.05	200139	O	Plasty		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
19	31.01.2023	0.05	200139	O	Plasty		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
20	31.01.2023	0.12	200102	O	Sklo		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
21	31.01.2023	0.12	200102	O	Sklo		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
22	13.02.2023	0.05	200101	O	Papír a lepenka		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
23	13.02.2023	0.05	200101	O	Papír a lepenka		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
24	13.02.2023	0.15	200102	O	Sklo		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
25	13.02.2023	0.15	200102	O	Sklo		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
26	13.02.2023	0.085	200139	O	Plasty		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.
27	13.02.2023	0.085	200139	O	Plasty		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
28	28.02.2023	0.22	200301	O	Směsný komunální odpad		A00	Produkce odpadu (vlastní vyprodukovaný odpad]	Infekční	INISOFT s.r.o.
29	28.02.2023	0.22	200301	O	Směsný komunální odpad		AN3	Předání jiné oprávněné osobě [kromě přepravce, dopravce)	Infekční	INISOFT s.r.o.

	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	IČO provozovny	IČP/IČZ/IČOB provozovny	Město (provozovna)	Ulice (provozovna)	Číslo popisné (provozovna)	Číslo orientační (provozovna)	PSČ (provozovna)	Název PSČ (provozovna)	ZÚJ (provozovna)	Název ZÚJ (provozovna)	Jméno kontaktní osoby	Příjmení kontaktní osoby	Telefonní číslo kontaktní oso
2	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
3	25417657	25417657-001	Liberec	Rumjancevova	696				556904	Liberec (nečleněné město)			
4	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
5	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
6	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
7	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
8	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
9	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
10	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
11	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
12	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
13	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
14	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
15	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
16	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
17	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
18	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
19	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
20	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
21	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
22	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
23	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
24	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
25	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
26	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
27	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
28	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			
29	25417657	1001020304	Liberec	Rumjancevova	696		46001	Liberec 1	556904	Liberec (nečleněné město)			

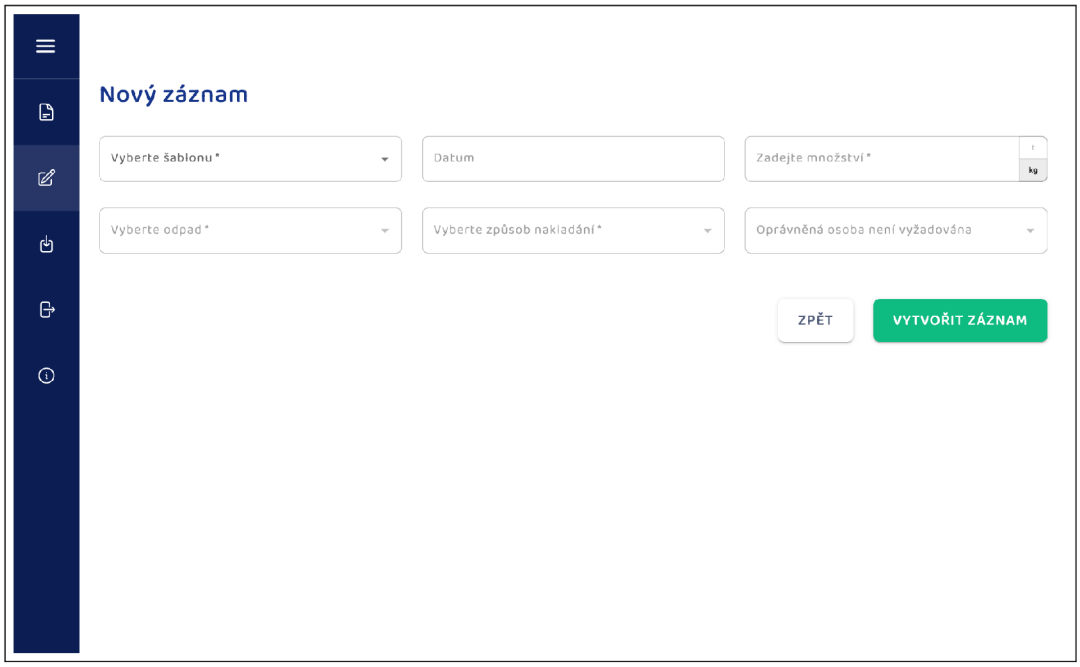
	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK
1	Email kontaktní osoby	Název partnera	IČO partnera	IČP/IČZ/IČOB partnera	Město (partner)	Ulice (partner)	Číslo popisné (partner)	Číslo orientační (partner)	PSČ (partner)	Název PSČ (partner)	ZÚJ (partner)	Název ZÚJ (partner)	Kód typu partnera	Název typu partnera
2		Marius Pedersen	42194920	CZA00218	Praha	Pražská	2	čo			500089	Praha 2	1	Právnická či fyzická osoba oprávněná k
3		Marius Pedersen	25417657	CZH00001	Hradec Králové	Průběžná			10400	Praha 114	500089	Praha 2	1	Právnická či fyzická osoba oprávněná k
4		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
5														
6											556904	Liberec (nečleněné město)	3	Občan obce
7		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
8		STATUTÁRNÍ MĚSTO LIBEREC	262978	1005527024	Liberec	Frýdlantská	183	2	46001	Liberec 1	556904	Liberec (nečleněné město)	4	Obec
9		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
10														
11		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
12														
13		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
14		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
15														
16														
17		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
18														
19		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
20														
21		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
22														
23		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
24		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
25		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
26														
27		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
28		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k
29														
30		Marius Pedersen	42194920	CZA12345	Hradec Králové	Průběžná	1	2	50002	Hradec Králové 2	569810	Hradec Králové	1	Právnická či fyzická osoba oprávněná k

Export ve formátu XLSX

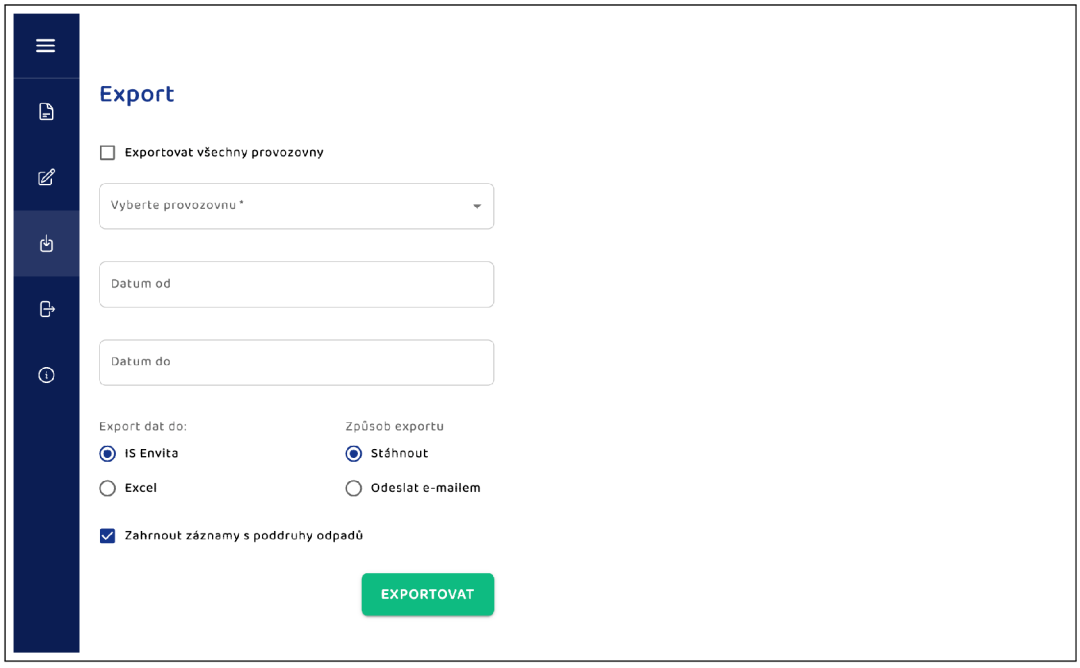
```xml
<IniTransfer xmlns="ISEnvita/Import" Version="1.0.0.0">
  <Header>
    <Version>1.0.0.7</Version>
    <NazevZdroje>JISPEN</NazevZdroje>
    <IdentifikatorExportu>13.02.2023 06:44:31</IdentifikatorExportu>
    <Sign>grusman@inisoft.cz</Sign>
  </Header>
  <DataCZ>
    <Subjekty>
      <Subjekt Id="MC_25417657_1001020304">
        <SubjektTypCZPO>
          <Identifikator>25417657</Identifikator>
          <SubjektNazev>INISOFT s.r.o.</SubjektNazev>
          <ProvozovnaKod>1001020304</ProvozovnaKod>
          <ProvozovnaNazev>INISOFT s.r.o.</ProvozovnaNazev>
          <VykazovaciKod>1001020304</VykazovaciKod>
          <CinnostNaUzemi>false</CinnostNaUzemi>
          <Adresy>
            <AdresaSidlo>
              <Ulice>Rumjancevova</Ulice>
              <CisloPopisne>696</CisloPopisne>
              <Obec>Liberec</Obec>
              <PSC>46001</PSC>
              <ZUJ>556904</ZUJ>
            </AdresaSidlo>
            <AdresaProvoz>
              <Ulice>Rumjancevova</Ulice>
              <CisloPopisne>696</CisloPopisne>
              <Obec>Liberec</Obec>
              <PSC>46001</PSC>
              <ZUJ>556904</ZUJ>
            </AdresaProvoz>
          </Adresy>
        </SubjektTypCZPO>
      </Subjekt>
      <Subjekt Id="WC_42194920_CZA12345">
        <SubjektTypCZPO>
          <Identifikator>42194920</Identifikator>
          <SubjektNazev>Marius Pedersen</SubjektNazev>
          <ProvozovnaKod>CZA12345</ProvozovnaKod>
          <ProvozovnaNazev>Marius Pedersen</ProvozovnaNazev>
          <VykazovaciKod>CZA12345</VykazovaciKod>
          <CinnostNaUzemi>false</CinnostNaUzemi>
          <Adresy>
            <AdresaSidlo>
              <Ulice>Průběžná</Ulice>
              <CisloPopisne>1</CisloPopisne>
              <CisloEvidencni>2</CisloEvidencni>
              <Obec>Hradec Králové</Obec>
              <PSC>50002</PSC>
              <ZUJ>569810</ZUJ>
            </AdresaSidlo>
            <AdresaProvoz>
              <Ulice>Průběžná</Ulice>
              <CisloPopisne>1</CisloPopisne>
              <CisloEvidencni>2</CisloEvidencni>
              <Obec>Hradec Králové</Obec>
              <PSC>50002</PSC>
              <ZUJ>569810</ZUJ>
            </AdresaProvoz>
          </Adresy>
        </SubjektTypCZPO>
      </Subjekt>
      <Subjekt Id="WC_34">
        <SubjektTypCZOO>
          <Adresy>
            <AdresaSidlo>
              <SidloAdresaZUJ>556904</SidloAdresaZUJ>
            </AdresaSidlo>
          </Adresy>
        </SubjektTypCZOO>
      </Subjekt>
    </Subjekty>
    <Odpady>
      <Odpad Id="1">
        <IdSubjektEvident>MC_25417657_1001020304</IdSubjektEvident>
        <Datum>2023-02-02</Datum>
        <Mnozstvi>0.002</Mnozstvi>
        <KatalogKod>180103</KatalogKod>
        <Kategorie>N</Kategorie>
        <KodNakladaniKod>AN3</KodNakladaniKod>
        <IdSubjektPartner>WC_42194920_CZA12345</IdSubjektPartner>
      </Odpad>
      <Odpad Id="2">
        <IdSubjektEvident>MC_25417657_1001020304</IdSubjektEvident>
        <Datum>2023-02-02</Datum>
        <Mnozstvi>0.002</Mnozstvi>
        <KatalogKod>180103</KatalogKod>
        <Kategorie>N</Kategorie>
        <KodNakladaniKod>A00</KodNakladaniKod>
      </Odpad>
      <Odpad Id="3">
        <IdSubjektEvident>MC_25417657_1001020304</IdSubjektEvident>
        <Datum>2023-02-03</Datum>
        <Mnozstvi>0.0005</Mnozstvi>
        <KatalogKod>180101</KatalogKod>
        <Kategorie>O</Kategorie>
        <KodNakladaniKod>BN30</KodNakladaniKod>
        <IdSubjektPartner>WC_34</IdSubjektPartner>
      </Odpad>
    </Odpady>
  </DataCZ>
</IniTransfer>
```


Export ve formátu XML

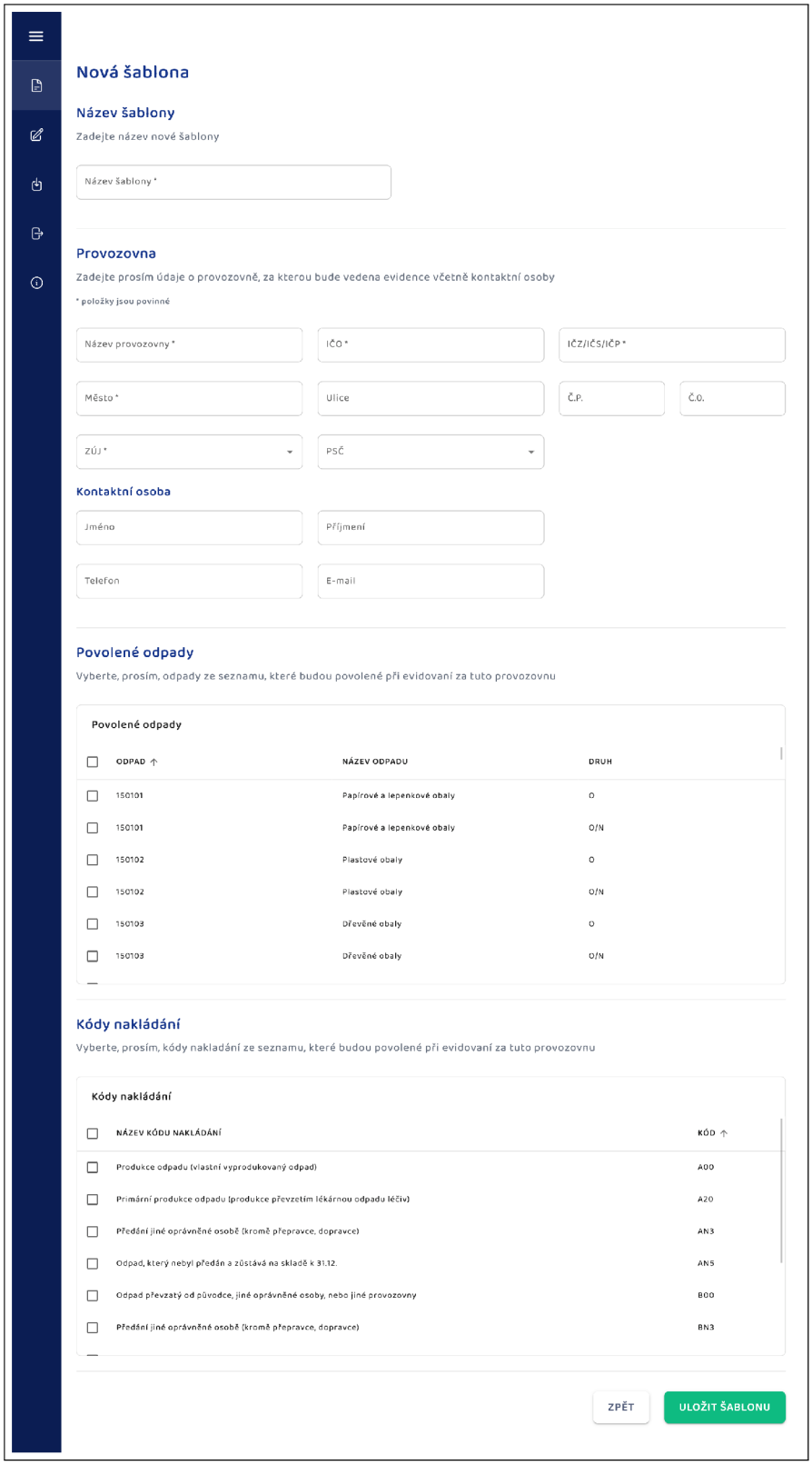
A.2 UKÁZKA UŽIVATELSKÉHO ROZHRANÍ



Stránka vytvoření nové evidence



Stránka exportu



Stránka vytvoření nové šablony

A.3 ODKAZ NA ZDROJOVÉ KÓDY

https://gitlab.tul.cz/nikita.tashchilin/jispen