VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM PRO SLEDOVÁNÍ VÝROBY

INFORMATION SYSTEM FOR PRODUCTION MONITORING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **ONDŘEJ JANOTA**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. ZBYNĚK KŘIVKA, Ph.D.**
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/21204/2017/xjanot02

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Janota Ondřej**
Obor: Informační technologie
Téma: **Informační systém pro sledování výroby**
Information System for Production Monitoring
Kategorie: Informační systémy

Pokyny:

1. Seznamte se s výrobními procesy firmy Fatra, a.s., a odpovídající terminologií.
2. Dle konzultací se zaměstnanci firmy proveďte analýzu a specifikaci požadavků na celý systém (včetně sledování plnění výrobních úkolů, spotřebu materiálu, vedení statistik produktivity).
3. Dle konzultací s vedoucím proveďte návrh systému včetně výběru implementačních technologií.
4. Návrh implementujte a proveďte nasazení systému do zkušebního provozu.
5. Ke zkušebnímu provozu získejte zpětnou vazbu a na jejím základě navrhněte další vývoj systému.

Literatura:

- Novák, J. a kol.: Organizace a řízení [skripta]. VŠB-TU Ostrava, 2007.
- dokumentace technologických postupů firmy Fatra, a.s.
- dále dle pokynů vedoucího (literatura k vybrané implementační technologii)

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Křivka Zbyněk, Ing., Ph.D.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

L.S.

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Hlavním cílem této bakalářské práce je navrhnout a vytvořit informační systém pro sledování výroby. Webová aplikace vytvořená pomocí nástroje Laravel bude sloužit pro evidování výroby a pracovní činnosti zaměstnanců oddělení paropropustných fólií firmy Fatra, a.s. Aplikace umožňuje efektivní vedení záznamů o výrobcích a automatické vedení záznamů o pracovní činnosti. Vedoucí mají možnost vytvářet záznamy o zakázkách podle uložených vzorů. V textu práce je popsán proces návrhu a vývoje aplikace. Výsledkem této práce je plně funkční a v praxi nasazený informační systém, který umožňuje uživatelům zaznamenávat veškeré pracovní úkony a získávat přehledné statistiky.

Abstract

The aim of this bachelor thesis is to design and implement an information system for production monitoring. The web application was created with Laravel framework. It will be used for recording production and work activities of employees that work in breathable films department at Fatra, a.s. The application allows the effective management of production records and the automatic creation of work activity records. This thesis describes the design and development of such application. The result of this work is fully functional and deployed information system that allows to record all work tasks and to generate clear statistics.

Klíčová slova

Informační systém, sledování výroby, evidence výroby, PHP, MySQL, JavaScript, Bootstrap, Apache, XAMPP, Laravel, webové aplikace

Keywords

Information system, production monitoring, production evidence, PHP, MySQL, JavaScript, Bootstrap, Apache, XAMPP, Laravel, web applications

Citace

JANOTA, Ondřej. *Informační systém pro sledování výroby*. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Zbyněk Křivka, Ph.D.

Informační systém pro sledování výroby

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doktora Zbyňka Křivky. Informace pro vytvoření jádra aplikace mi poskytla vedoucí firemního oddělení paní Jarmila Mlýnková. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Ondřej Janota
15. května 2018

Poděkování

Tímto bych velmi rád poděkoval vedoucímu práce panu Ing. Zbyňku Křivkovi, Ph.D. za vedení mé bakalářské práce, odbornou pomoc a cenné rady. Také bych chtěl poděkovat zaměstnancům firmy Fatra, a.s. za jejich spolupráci a poskytnutí informací.

Obsah

Kapitola 1

Úvod

Informační systémy jsou dnes již velmi používanou metodou pro zpřehlednění a zjednodušení každodenní práce. Systém lépe uchovává informace, se kterými je nutno dále pracovat. Zefektivňuje výslednou práci, jelikož provádí mnoho úkonů, které bychom museli zdlouhavě dělat sami ručně. Je proto nutné zaměřit se na požadavky klienta, aby práce uživatelů se systémem byla mnohonásobně jednodušší a efektivnější, než-li je tomu doposud bez systému. Tato práce je věnována návrhu a implementaci informačního systému pro sledování a evidenci výroby.

Cílem této práce je vytvoření webové aplikace, která zjednoduší evidenci výroby a umožní získání zajímavých statistik oddělení paropropustných fólií firmy Fatra, a.s. Aby nebylo potřeba zdlouhavého školení uživatelů, je nutné, aby aplikace poskytovala přehledné a efektivní rozhraní. V systému bude možné spravovat prvky důležité k realizaci jednotlivých zakázek výroby, systém také umožní uchovat informace o práci se součástmi stroje nutné k získání statistik pracovní činnosti. Aplikace navíc bude ukládat informace o provedené práci jednotlivých pracovníků a poskytovat tak vedení oddělení informace nutné k výpočtu výplat.

Práce se zabývá specifikací požadavků na výsledný systém, které jsou poté analyzovány. Z provedené analýzy je navržena architektura aplikace a vybrána technologie pro tvorbu aplikace.

Ve druhé kapitole se zaměříme na charakteristiku tvorby webových aplikací. Podíváme se na existující programovací jazyky používané při tvorbě webových aplikací spolu se základními technologiemi a knihovnami pro usnadnění práce. Také se podíváme na jednotlivé fáze vývoje informačního systému.

V další kapitole se detailněji seznámíme s výrobními procesy, probereme jejich jednotlivé části a jejich provedení. Rozebereme vybrané jazyky a technologie, vhodnost jejich využití a případně navrhneme lepší řešení, které je možné využít při vylepšování a rozšiřování systému.

Ve čtvrté kapitole se zaměříme na výběr technologií a implementaci. Co vše a jakým způsobem bylo vyvíjeno je popsáno v kapitole 5. Kapitola 6. se pak zabývá testováním výsledné aplikace. Na rozšíření, která by mohla býti v budoucnu vytvářena, se podíváme v sedmé kapitole. V poslední kapitole se budeme zabývat zhodnocením provedené práce.

1.1 O firmě Fatra, a.s.

Firma patří mezi významné světové zpracovatele plastů (PVC[1], PE[2] a PET[3]). Je nedílnou součástí plastikářského průmyslu. V roce 2016 více jak 75 % produkce směrovalo na zahraniční trhy. Provozuje moderní technologie na zpracování plastů ve výrobních centech v Napajedlích a Chropyni. Prodává své výrobky do více než 50 zemí světa. Mezi zpracovávané suroviny patří PVC, PET či PP[4]). Je členem koncernu Agrofert, který sdružuje více než 250 významných subjektů ze sektoru chemie, zemědělství, potravinářství a médií. Mimo jiné při vývoji nových výrobků Fatra, a.s. spolupracuje s vysokými školami (Univerzita Tomáše Bati ve Zlíně a Univerzita Pardubice) a vývojovými pracovišti [4].

[1] Polyvinylchlorid - jedna z nejpoužívanějších umělých hmot
[2] Polyethylen - termoplast vznikající polymerací ethenu
[3] Polyethylentereftalát - termoplast ze skupiny polyesterů
[4] Polypropylen - termoplastický polymer ze skupiny polyolefinů

Kapitola 2

Tvorba webové aplikace

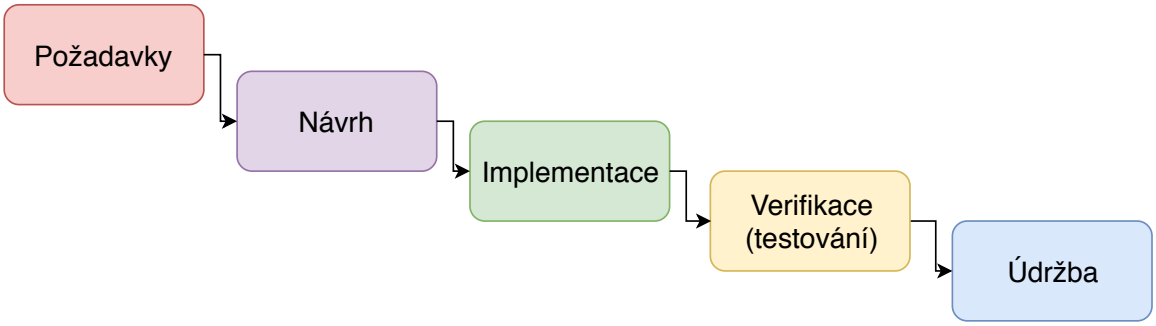
V dnešní době jsou webové aplikace nedílnou součástí našeho života. Protože jsou velmi rozšířené a neustále se rozšiřují možnosti pro tvorbu aplikací, existuje obrovské množství programovacích nebo jiných jazyků, díky kterým můžeme jednodušeji vytvářet webové aplikace. Jelikož existuje mnoho vývojářů webových aplikací, nalezneme mnoho volně dostupných užitečných nástrojů a knihoven, které můžeme při vývoji aplikací použít pro dosažení nejlepších, případně chtěných, výsledků.

Webové aplikace jsou daleko univerzálnější, než-li je tomu tak u desktopových či mobilních aplikací, zabývá se jimi čím dál více vývojářů. Hlavní výhodou je, že uživatelé pro použití takových aplikací nemusí kromě webového prohlížeče instalovat žádný další software. Dnes již lze webové aplikace používat stejným způsobem jak na počítačích, tak i na mobilních zařízeních nebo tabletech. Práce vývojářů takových aplikací je mnohem jednodušší, protože odpadá složité řešení jednotlivých platforem. I v případě webových aplikací je nutno přihlížet na různé desktopové prohlížeče či prohlížeče v mobilních zařízeních, mnoho frameworků však s responzivitou v různých prohlížečích zapracovalo a částečně ušetřilo vývojářům práci.

Pro tuto práci však byl požadavek vytvořit webovou aplikaci pouze pro prohlížeče desktopové. S použitím vybraných technologií lze s velmi malými úpravami v případě rozšíření systému počítat i s prohlížeči např. v mobilních zařízeních.

2.1 Fáze vývoje systému

Při vývoji informačního systému jsem postupoval chronologicky podle vodopádového modelu 2.1.



Obrázek 2.1: Vodopádový model

Nejprve bylo potřeba sepsat požadavky klienta a získat tak představu o vyvíjeném systému. Bylo také nutné zanalyzovat problematiku evidence výroby.

Další fází byl výběr webových technologií. Tento výběr je popsán v kapitole 4. Po výběru technologií přišel na řadu návrh systému. Po jeho dokončení následovala implementace, kterou odstartovala tvorba databáze na základě získaných poznatků z předchozí fáze vývoje. Následně byla naprogramována webová aplikace.

V předposlední fázi bylo provedeno testování aplikace a získání zpětné vazby od klienta. Možná vylepšení a rozšíření systému byla hlavní myšlenkou poslední fáze.

Kapitola 3

Analýza

Nejdůležitějším procesem při vývoji informačního systému je jeho analýza. Další vývoj je na tomto procesu plně závislý, proto je potřeba analýzu provádět důkladně. Účelem analýzy je zaznamenat všechna potřebná data a probíhající procesy ve firmě.

Nejprve jsem se sešel s klientem, se kterým jsem probral všechny náležitosti systému a důkladně prošel technologické postupy. Jelikož na oddělení pracuji, s postupy prováděnými pracovníky jsem byl již dříve seznámen. Bylo nutné porozumět postupům, které provádí vedení oddělení.

3.1 Problematika evidence výroby

Evidence výroby je velmi široký pojem, jenž zahrnuje správu polotovarů a výrobků. Také je potřeba evidovat výrobu samotnou - zakázky a jejich plnění. U tohoto oddělení je nutné také evidovat použití jednotlivých částí stroje a také evidovat používání barev. Díky informačnímu systému je taková evidence mnohem přehlednější a snazší. Také se zde otevírá možnost jednoduchého získávání veškerých důležitých statistik.

3.1.1 Existující řešení

Oddělení PPF (paropropustné fólie) ve firmě Fatra, a.s. nevlastní žádný funkční informační systém, který by evidoval výrobu a procesy s ní spojené. Existuje však systém využívající Microsoft Office Excel. Tento systém byl vytvářen před mnoha lety a dnes již obsahuje mnoho nepotřebných informací. Z pohledu pracovníka z hlediska použitelnosti je neefektivní a nepřehledný. To přispělo k mé motivaci vytvořit nový systém, který plně nahradí systém dosavadní.

I přes to, že je systém velmi nepřehledný, obsahuje informace potřebné k vývoji systému nového. To zásadně snížilo počet konzultací s klientem pro získávání informací, jak mají jednotlivé části systému fungovat. Dosavadní systém tak přispívá k pochopení vnitřní logiky jednotlivých procesů.

3.1.2 Diagram případů užití

Diagram případů užití (Use Case Diagram) se používá k popisu chování systému z hlediska uživatele a zachycuje, které typy uživatelů se systémem pracují a jaké činnosti v rámci systému vykonávají[10]. Diagram je rozdělen na 2 části. V první části (viz obr. 3.1) jsou vyobrazeny role „Admin“ a „Vedoucí“. Ve druhé části (viz obr. 3.2) přibyla role „Pracovník“,

která nemá práva přímo pracovat se zakázkami, má však právo nepřímo upravovat plnění zakázek vytvářením nových výrobků.

Návrh

Protože bude pro firmu Fatra, a.s. později vytvářeno rozšíření systému, byl vytvořen diagram případů užití se čtyřmi aktéry, pro nynější verzi systému by postačily role tři. Prvním takovým je *pracovník*, který má omezený přístup v systému z důvodu ochrany citlivých dat o uživatelích a informacích o zakázkách. Druhým aktérem je *vedoucí*, který navíc spravuje údaje o zaměstnancích. Může měnit, vyhledávat a mazat i citlivé údaje, zadávat potřebné informace o výrobě apod. Třetím aktérem je *mistr/instruktor* - tento aktér bude potřeba při rozšíření systému, nyní má však stejná práva jako vedoucí. Poslední, čtvrtý, aktér *admin* má v nynějším systému stejně důležitou roli jako vedoucí z důvodu jednoduššího zaučování používání systému (v rozšířeném systému je očekávána jako významnější role pro řízení uživatelských účtů). Pouze admin může vytvořit nového admina.

3.1.3 Zakázka

Velmi důležitým požadavkem se stala práce se samotnými zakázkami. Bylo nutné, aby se zakázky daly jednoduše vytvářet a upravovat. Také bylo nutno kontrolovat jejich aktuální plnění a případně měnit jejich pořadí. První diagram 3.1 zobrazuje, jakým způsobem uživatel zasahuje do systému při práci se zakázkami.

Stručný výstup analýzy

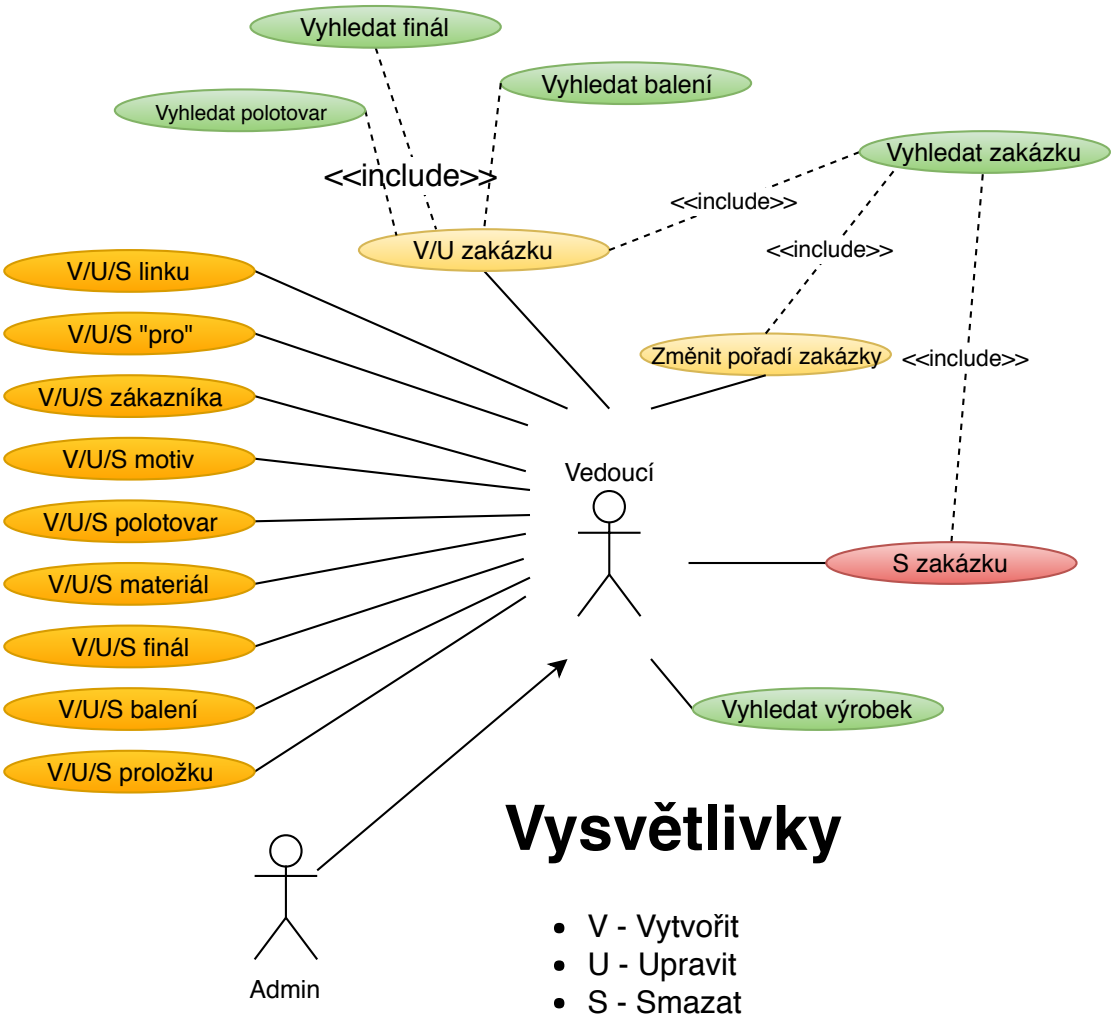
Detailní výstup analýzy požadavku si můžete prohlédnout buď v následujícím seznamu, nebo na obrázku 3.1.

- Přístup k provádění akcí musí být zabezpečen před vstupem neoprávněných osob.
- Uživatel se před používáním systému musí přihlásit.
- Systém musí uchovávat informace o linkách, zákaznících, motivech, polotovarech, finálech, baleních a o tom, pro koho jsou výrobky vyráběny.
- Systém umožní jednoduché vytváření, úpravu a mazání všech částí zakázky.
- Vedoucí musí mít možnost vytvořit, upravit i smazat jednotlivé části zakázek.
- Systém musí zabránit smazání jednotlivých částí při využití jakoukoli zakázkou pro udržení konzistence databáze.

3.1.4 Ostatní části systému

Druhá část analýzy se již zabývá i používáním systému rolí „Pracovník“. V tomto případě bylo nutné zaměřit se na používání systému pracovníkem, který má nejvíce omezená práva v systému. I přes toto omezení dokáže nepřímo upravovat data spojená se zakázkami, a to prací s výrobky. Dále také zaznamenává stavy odpadů na oddělení, práci s částmi linek a případné závady.

Požadovaný systém byl rozšířen o evidenci práce zaměstnanců a sklad barev a ředidel. Ke konci měsíce musí vytvořit záznamy o docházce do týdeníčků (tyto záznamy systém automaticky přepíše v den změny) jako výpomoc při vytváření výplat.



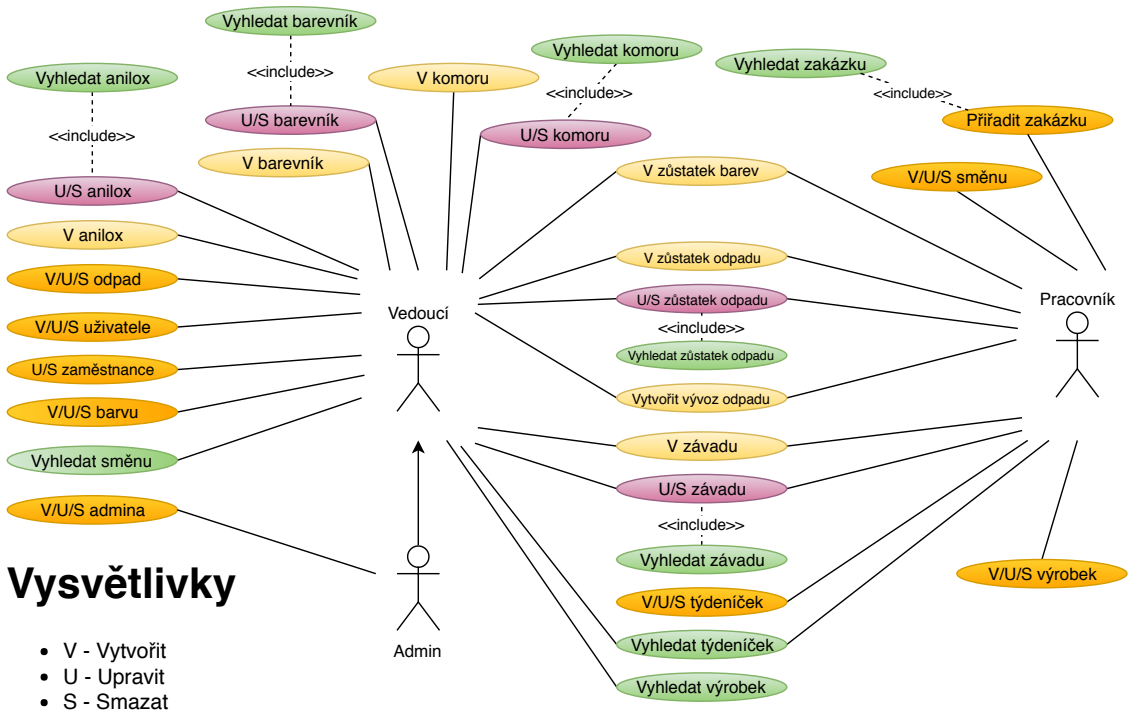
Obrázek 3.1: Diagram případu užití - část Zakázka

Stručný výstup analýzy

Opět si níže můžeme prohlédnout detailní výstup analýzy v následujícím seznamu nebo na obrázku 3.2.

- Přístup k provádění akcí musí být zabezpečen před vstupem neoprávněných osob.
- Uživatel se před používáním systému musí přihlásit.
- Systém musí uchovávat informace o odpadech, aniloxech, barevnících, raklových komorách, barvách a ředidlech, týdeníčcích, směnách, závadách a výrobcích.
- Systém musí umožnit vytvářet, upravovat a mazat výše uvedené informace.
- Systém musí umožňovat vyhledávat všechny uchované informace.
- Systém také musí uchovávat informace o uživatelích a zaměstnancích.

- Pracovníkovi musí být umožněno předčasně vytvářet záznamy o pracovní činnosti ve formě týdeníčků.
- Systém musí dokázat upravit dříve vytvořené týdeníčky.
- Vedoucímu musí být umožněn přístup ke všem datům vytvářeným pracovníkem.



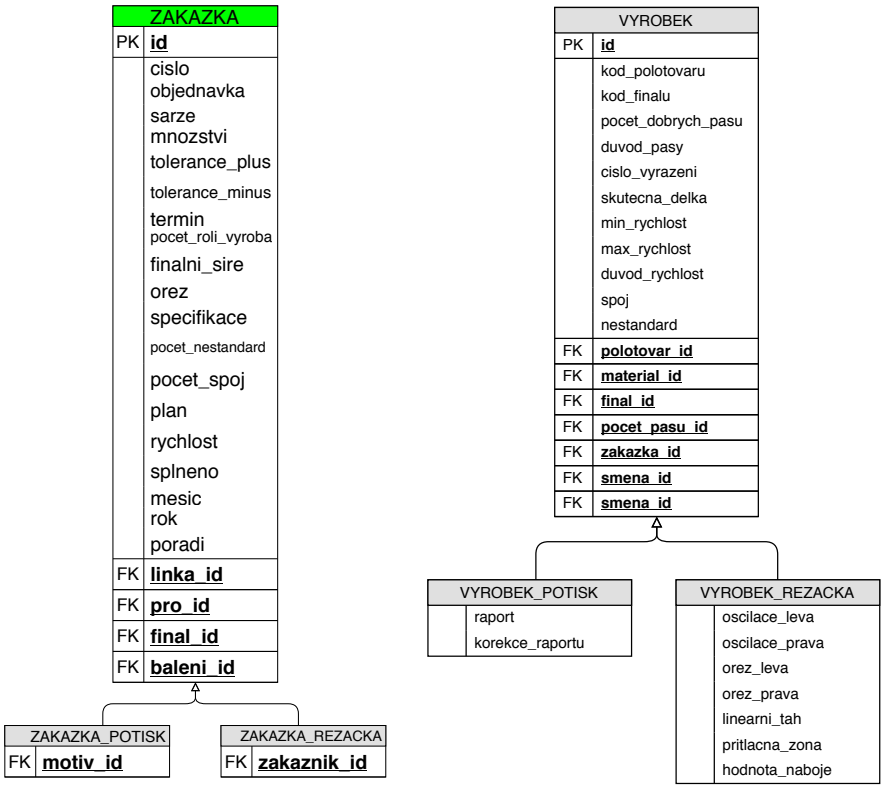
Obrázek 3.2: Diagram případu užití - ostatní části

3.2 ER diagram

ER diagram jsem rozdělil na čtyři části - Zakázku (viz obr. A.1 v příloze), Vývozy (viz obr. A.3 v příloze), Operace (viz obr. A.2 v příloze) a Týdeníček (viz obr. A.4 v příloze), aby byl přehlednější (společné entity jsou barevně označeny). Při návrhu bylo nutné dbát na správný výběr všech entit a jejich atributů. Také bylo potřeba zvolit nejvhodnější datové typy každého atributu tak, aby zabíral co nejméně místa, ale také, aby pokryl všechny požadované hodnoty.

3.2.1 Zakázka

Pomocí speciálního vztahu generalizace (zobecnění) byly pro entitu Zakázka vytvořeny dvě tabulky (viz levá část obr. 3.3), které obsahují jediný atribut, jenž jednoznačně rozlišuje zakázku podle toho, na které lince je prováděna. Linka je rozlišena podle typu pro potisk nebo řezání. V případě potisku se zakázka provádí podle určitého motivu, u řezání se pak provádí pro určitého zákazníka.



Obrázek 3.3: Schéma generalizace pro entitu Zakázka a Výrobek

Také pro entitu Výrobek bylo potřeba vytvořit pomocí vztahu generalizace další dvě tabulky (viz pravá část obr. 3.3). Zde je opět třeba rozlišit, na jaké lince byl výrobek vytvořen. U potiskovacích linek se může zapisovat raport a jeho korekce, u řezačky pak možnost zapsat u každého výrobku informace o ořezu.

3.2.2 Vývozy

Do ER diagramu A.3 byla přidána evidence skladu barev a ředidel. Toto rozšíření velice pomáhá pracovníkům, jelikož v dřívějším systému prováděli evidenci pouze z jediného stolního počítače, což bylo velmi nepraktické, protože musel odepsané barvy přepisovat např. na papír, aby jej později mohl na PC zaevidovat.

3.2.3 Týdeníček

Velkou výhodou se stal týdeníček, který je popsán v ER diagramu A.4. Týdeníček slouží k zaznamenání pracovní činnosti zaměstnance, respektive kolik hodin pracoval na nějaké lince a na kolik procent plnil plán směny. Pokud je zaměstnanec nemocný nebo si vzal volno, musí sám vytvořit týdeníček s počtem odpovídajících hodin. Přidáním této části se velice zautomatizovalo evidování provedené práce a ušetřila se i práce pracovníkům.

Kapitola 4

Implementace

Implementace se v této práci zabývá vytvořením aplikace podle dříve definovaných diagramů. Obecně by se dalo tvrdit, že implementace patří k jednodušším fázím vývoje, protože se řídíme podle přesně definovaných diagramů. Naproti tomu se však staví fakt, že vytvoření webové aplikace je velmi časově náročné, musí se počítat např. s návrhem designu a testováním.

Při práci je vhodné využít užitečné nástroje, které obsahují mnoho funkcí, jež mají za úkol tuto práci usnadnit.

4.1 Vybrané technologie

Po analýze požadavků byly zvoleny webové technologie, jež jsou výhodné pro jejich nezávislost na platformě. Taktéž se stává výhodou to, že není potřeba instalovat žádný dodatečný software, ale postačí pouze obyčejný prohlížeč. Jedinou nevýhodou může být optimalizace pro všechny dostupné prohlížeče a rozlišení. V případě této aplikace se přihlíží na použití v prohlížeči Internet Explorer, otestován je však také v prohlížečích Chrome či Mozilla Firefox.

Pro vývoj jsem použil následující technologie a nástroje, které jsou podrobněji popsány níže:

- HTML, CSS,
- nástroj Bootstrap,
- JavaScript, JQuery, Ajax,
- PHP, MySQL,
- nástroj Laravel
- a XAMPP.

4.1.1 HTML

HTML je zkratka anglického Hypertext Markup Language, tedy značkovací jazyk pro hypertext. Je to jazyk používaný pro tvorbu webových dokumentů. Jeho předností je univerzálnost, protože je použitelný na všech platformách. Ve skutečnosti jsou dokumenty HTML pouhými soubory uloženými např. v ASCII nebo jiných textových formátech. Pokud jsou soubory zpřístupněny online, lze k nim přistupovat odkudkoli na světe. Nezáleží na tom, zda uživatel používá PC, Mac či mobilní telefon.[11]

Jazykem HTML lze také upravovat vzhled stránek, to však není moc vhodné. Pro takové případy se používá pro úpravu vzhledu technologie CSS. Ačkoliv je HTML multiplatformní, pro kvalitní vzhled stránky je nutné přihlížet na jednotlivé platformy zvlášť.

4.1.2 CSS

CSS (Cascading Style Sheets) jsou kaskádové styly sloužící ke specifikaci formátování HTML stránky např. pomocí souboru *.css. Pokud je vše správně uděláno a CSS je použito konzistentně, je možné provádět s velmi malým úsilím (pouhou editací CSS souboru) vizuální změny webové stránky nebo jen některých požadovaných částí[3]. Na internetu je mnoho informací jak s CSS správně pracovat.

Samotné CSS vzniklo někdy kolem roku 1997 jako kolekce metod pro grafickou úpravu webových stránek. Slovo kaskádové zde znamená, že se na sebe mohou vrstvit definice stylu, ale platí pouze ta poslední.[7]

4.1.3 Framework Bootstrap

Bootstrap je volně dostupný nástroj pro vývoj webových stránek pomocí CSS a JS. Rychle navrhneme své nápady nebo vytvoříme celou aplikaci pomocí Bootstrap Sass proměnných a tříd, responzivní mřížkový systém, rozsáhlých přednastavených komponent a také výkonných zásuvných modulů v JQuery.

Bootstrap vznikl ve společnosti Twitter v roce 2010. Než se stal Bootstrap volně dostupným, byl znám jako Twitter Blueprint. Ve firmě se více než rok před zveřejněním začal používat jako příručka pro vývoj interních nástrojů.

Původně vyšel 19. 8. 2011, od té doby bylo vytvořeno více než 20 verzí, včetně dvou hlavních přepisů s v2 a v3.

Bootstrap 4 zohledňuje dvě klíčové změny - migraci na Sass a přesun do flexboxu CSS. Záměrem vývojářů je pomoci posunout komunitu pro vývoj webových aplikací vpřed tím, že budou prosazovat novější vlastnosti CSS, nové technologie v modernějších prohlížečích a méně závislostí.[1].

Pro implementaci jsem využil základní verzi Bootstrap, ten definuje mnoho interaktivních prvků jako tlačítka, menu a další elementy. Nejnovější verzi Bootstrap lze získat stáhnutím zkompilovaných souborů CSS a JavaScript na stránkách vývojářů[1].

4.1.4 JavaScript

JavaScript (JS) je interpretovaný programovací jazyk se základními objektově orientovanými schopnostmi. Univerzální jádro jazyka bylo vloženo do mnoho webových prohlížečů. Pro webové programování bylo rozšířeno přidáním objektů reprezentujících okno webového

[1] Dostupné na www.getbootstrap.com

prohlížeče a jeho obsah. Klientská verze JS umožňuje vložit do webových stránek proveditelný obsah, což znamená, že ze statických HTML dokumentů se můžou stát dokumenty dynamické - prohlížeč dynamicky vytváří obsah HTML v reálném čase.

Svou syntaxí připomíná jazyky C, C++ a Javu, podobnost však syntaxí končí. JS je programovací jazyk bez typové kontroly, proměnné tak nemusí mít specifikovaný typ. Objekty se podobají spíše asociativním polím jazyka Perl. Dědičnost objektové orientace odpovídá prvkům jazyků Self a NetworkScript a výrazně se tak liší od dědičnosti v jazycích C++ a Javě. V mnoha ohledech JavaScript přebírá myšlenky jazyka Perl (např. regulární výrazy či práce s poli).

JavaScript není Javou, i když s ní tvoří velice dobrý tým. Java na rozdíl od JS nemá žádnou kontrolu nad prohlížečem, naopak však může vytvářet grafiku, pracovat se sítí a v režimu více vláken.[5]

Aby JS fungoval, je nutné jej povolit v prohlížeči. Zda-li je použit, poznáme např. podle importovaného souboru s koncovkou *.js*.

4.1.5 JQuery

Jedná se o knihovnu JavaScriptu, která napomáhá vývojářům s běžnými úkoly a zjednodušuje úkoly složitější. Je velmi oblíbená, jelikož má schopnost pomáhat se širokou škálou úkolů. Poskytuje víceúčelovou abstraktní vrstvu pro běžné webové skriptování.

S JQuery můžeme např. jednoduše přistupovat k jednotlivým elementům dokumentu HTML pomocí selektorů (u JavaScriptu je potřeba procházet strom modelu DOM[2] a vyhledat v něm tyto elementy). Dále umí upravovat vzhled webové stránky, měnit obsah dokumentu, reagovat na akce uživatele nebo animovat změny v dokumentu. Navíc také lze načítat data ze serveru bez nutnosti obnovení stránky (např. pomocí technologie Ajax).[2]

JQuery je velmi rozsáhlá knihovna, která je volně dostupná[3].

4.1.6 Ajax

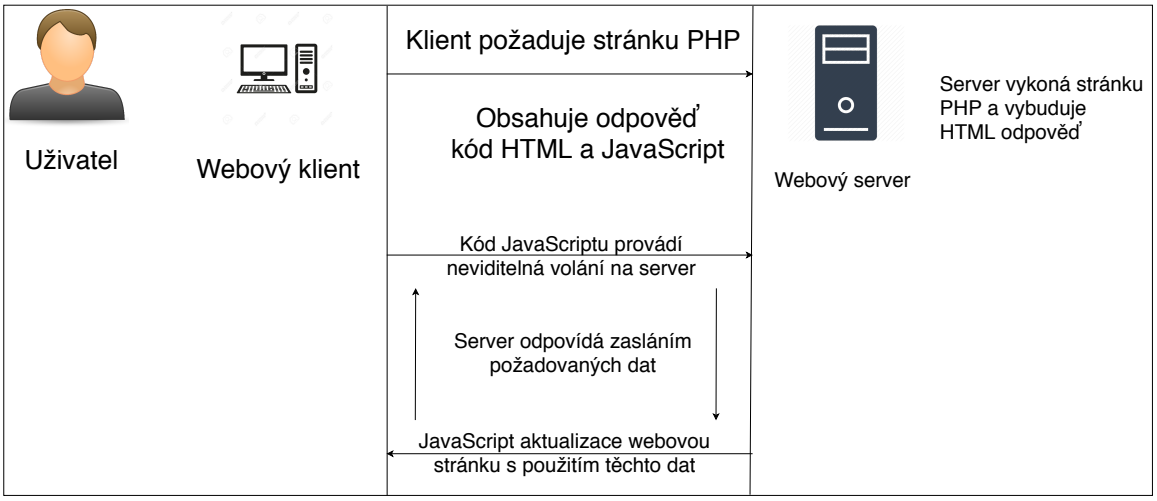
V roce 2005 přišel Jesse James Garret s novinkou zvanou Asynchronous JavaScript and XML (asynchronní JS a XML). Od vzniku této technologie již reprezentoval řadu věcí, jelikož je tímto termínem označováno spousta funkcí a technik. Na základní úrovni zahrnuje ajaxová aplikace tyto technologie:

- kód jazyka JavaScript pro zachytávání uživatelských interakcí a jiných událostí webového prohlížeče,
- objekt *XMLHttpRequest*, jenž umožňuje odesílat požadavky na server bez přerušení ostatních úloh webového prohlížeče,
- soubor na serveru poskytující data ve formátu XML, HTML nebo JSON,
- a další kód jazyka JS pro interpretování dat ze serveru a jejich prezentaci na stránce.

Podstata Ajaxu spočívá v tom, že JavaScriptu na straně klienta je umožněno volat v pozadí server a podle potřeby tak získávat potřebná data. Tímto způsobem je možné aktualizovat některé části stránky bez nutnosti opětovně načítat stránku.[3]

Na obrázku 4.1 je reprezentace toho, co se děje, když je typická webová stránka s technologií AJAX vyžádána jejím návštěvníkem.

[2] Document Object Model
[3] www.jquery.com



Obrázek 4.1: Typické volání AJAXu

Výhodou Ajaxu, oproti JavaScriptu, je asynchronní provádění kódu. To znamená, že po odeslání požadavku protokolu HTTP webový prohlížeč nečeká na odpověď, ale začne provádět další příkazy skriptu. Později, až prohlížeč obdrží odpověď ze serveru, začne ji dále zpracovávat.[2]

4.1.7 PHP

Zkratka PHP se původně interpretovala jako Personal Home Page (osobní domovská stránka) a označovala jazyk používaný především k realizaci formulářů používaných na webových stránkách. Plná verze byla označována jako PHP/FI[4]. V průběhu času se ustálil pojem Hypertext Preprocesor.[8]

Hlavními protihráči PHP jsou Perl, Microsoft Active Server Pages (ASP), Java Server Page (JSP) a Allaire ColdFusion. V porovnání s těmito produkty má PHP mnoho předností:

- vysoká výkonnost,
- rozhraní pro mnoho druhů databázových systémů,
- zabudované knihovny pro implementaci mnoha běžných webových úloh,
- nízké náklady,
- snadná výuka a použití,
- přenositelnost,
- zdrojový kód PHP je k dispozici.

PHP má schopnost připojovat se bez jakýchkoli prostředníků k mnoha databázovým systémům. Kromě MySQL se může mj. přímo připojit na PostgreSQL a Oracle.

Protože byl PHP již od počátku navrhován pro využití ve webových aplikacích, obsahuje mnoho funkcí, jež jsou určeny k plnění úkolů svázaných s webem. Můžeme tak generovat

[4]Personal Home Page Forms Interpreter

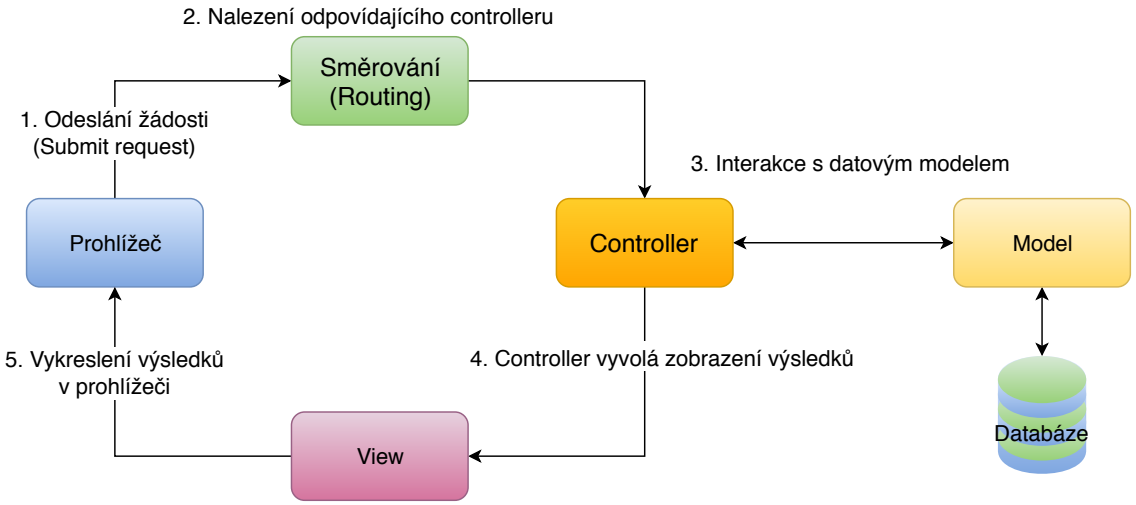
obrázky formátu GIF, připojovat se k různým síťovým službám, odesílat emaily, pracovat s cookies a nebo třeba generovat dokumenty formátu PDF.[12]

PHP je open source produkt, máme tak přístup k jeho zdrojovému kódu a můžeme jej používat, upravovat a dále distribuovat bez jakýchkoliv poplatků. Poslední dostupnou verzí je PHP 7.2[5].

4.1.8 Laravel

Laravel je framework pro webové aplikace s expresivní, elegantní syntaxí. Pokouší se ulehčit trápení při vývoji webových aplikací.

Je volně rozšířený PHP framework pro vývoj webových aplikací. Jako je tomu tak u JQuery, opět se jedná o otevřený systém pod licencí MIT. Laravel využívá architektury MVC (Model–View–Controller), kde *Model* jsou aplikační data a funkce, *View* je prezentace dat např. v HTML a *Controller* spravuje interakci mezi modelem a pohledem (viz obr. 4.2). Laravel je optimalizován pro reálný svět, disponuje tedy často užívanými procedurami, které jsou nutné pro vývoj webové aplikace.[13]



Obrázek 4.2: Proces zaslání požadavku podle architektury Laravel

Framework Laravel jsem vybral, jelikož je nejpoužívanějším frameworkem na světě. Dokazuje to také statistika vyhledáváných dotazů na Google Trends[6].

4.1.9 MySQL

MySQL je velmi rychlý a robustní relační databázový systém (RDBMS). Databáze umožňuje efektivně ukládat, hledat, řadit a získávat data. Server SQL se stará o to, aby k databázi mohlo přistupovat více uživatelů zároveň a zajišťuje, aby to byli pouze oprávnění uživatelé. MySQL je tedy víceuživatelský a více vláknový (multi-threaded) server. Používá SQL (Structured Query Language) - celosvětově používaný standardizovaný jazyk pro databáze. Je volně přístupný od roku 1996, jeho historie však sahá až do roku 1979. MySQL je sice k dispozici pod Open Source licencí, ale je v případě potřeby dostupná i jako komerční

[5]Dostupné na www.php.net

[6]Nejvyhledávanější dotazy na Google - https://goo.gl/r8cr9w

verze. Hlavními soupeři MySQL jsou PostgreSQL, Microsoft SQL server a Oracle. MySQL má mnoho předností:

- vysoká výkonnost,
- nízké náklady,
- snadná konfigurace a výuka,
- přenositelnost,
- zdrojový kód je dostupný.

MySQL je tedy k dispozici pod licencí Open Source a můžeme jej používat na mnoha různých UNIXech nebo i Microsoft Windows.[12]

4.1.10 XAMPP

XAMPP je volně šiřitelný program, multiplatformní doplněk pro spuštění serveru, který umožňuje rychlou a snadnou instalaci Apache, MySQL, PHP a Perl. Lze spustit jak na Windows, tak i Linux nebo OS X a je volně dostupný.[6]

Apache

Působí jako webový server. Jeho hlavní úlohou je nejen zpracování požadavků, které uživatelé odesílají prostřednictvím svých webových prohlížečů, ale také zobrazení výsledků připravených pomocí kódu umístěného ve vyžádaných souborech. Je to velmi výkonný stroj a může splnit prakticky všechna přání, jaká můžeme jako správce webu mít.[9]

Kapitola 5

Vývoj systému a aplikace

5.1 Tvorba databáze

V první řadě bylo potřeba vytvořit databázi MySQL, jež je naprogramována podle ER diagramu (složeného z jednotlivých částí). Bez takové databáze by nikdy informační systém nemohl vzniknout a správně fungovat.

Pro vytvoření databáze bylo nutné nadefinovat všechny tabulky a jejich atributy s výběrem ideálních datových typů. Tyto typy jsem přesněji určoval až v průběhu první fázi testování, které je popsáno v kapitole 6.

Aby vše fungovalo, musely se jednotlivé tabulky v databázi provázat pomocí cizích klíčů a zprovoznit databázi na serveru APACHE. Zprovoznění lze provést dvěma způsoby:

- spuštěním skriptu ve formátu *.sql*,
- použitím příkazu `php artisan migrate` v příkazové řádce pomocí Artisan[1].

5.2 Komunikace s databází

Abychom mohli pracovat se systémem, musíme mít přístup k datům v databázi - vytvářet, upravovat, mazat a získávat tato data. Takové operace lze provádět pomocí:

- Laravel Query Builder, který poskytuje pohodlné, plynulé rozhraní pro vytváření a spouštění databázových dotazů[2],
- nebo pomocí Laravel Eloquent, jenž poskytuje jednoduchou implementaci ActiveRecord pro práci s databází[3].

V drtivé většině případů jsem využil druhou možnost. Každá tabulka v databázi může mít svůj vlastní *Model*, který umožňuje dotazování dat a také vkládání nových záznamů do dané tabulky. Ne každý však může tyto operace provádět, a proto je potřeba řešit autorizaci uživatelů. Laravel opět poskytuje velmi snadné rozhraní, díky kterému máme informace o uživateli přístupné odkudkoli v aplikaci. Poté se pomocí jednoduchých podmínek ověřilo, zda je uživatel oprávněn přistupovat či manipulovat se záznamy v tabulkách. O tom, kdo smí a nesmí provádět určité operace, víme již z diagramů případu užití v kapitole 3.

[1] Rozhraní příkazového řádku, které je součástí nástroje Laravel, více informací na https://laravel.com/docs/5.6/artisan

[2] Více informací na https://laravel.com/docs/5.6/queries

[3] Více informací na https://laravel.com/docs/5.6/eloquent

5.2.1 Laravel Model

Model zastupuje důležitou roli v provázání jednotlivých záznamů v tabulkách databáze. Aplikací relací z ER diagramu se velmi zjednodušuje další komunikace s databází. V implicitní nastavení se všechny modely nachází ve složce *app* daného projektu.

Relace 1:1

Tuto jednoduchou relaci aplikujeme pomocí metod `belongsTo()` a `hasOne()`. Při použití první metody lze provázat záznamy pomocí funkce `associate()`, která zařídí propojení záznamů pomocí primárního klíče navazovaného záznamu. Naopak pomocí druhé metody získáváme přímý přístup z připojeného záznamu. Pokud chceme zrušit propojení, používáme funkci `dissociate()`.

Pro příklad přístupu využijeme vztah z ER diagramu A.4 mezi Uživatelem a Zaměstnancem: příkaz `$user->zamestnanec` nám vrátí záznam zaměstnance. Pokud neexistuje žádné provázání, vrátí se `NULL`.

Relace 1:N

Zde se používají stejné metody pro propojení záznamů jako u relace 1:1. Oproti této relaci se pouze změní funkce pro získání přístupu z propojeného záznamu a to `hasMany()`. Pokud použijeme tuto funkci, nezískáme pouze jediný záznam, ale pole všech propojených záznamů.

Pro příklad přístupu využijeme vztah z ER diagramu mezi Linkou a Zakázkou - *1:N*: příkaz `$linka->zakazka` vrátí pole propojených záznamů. Při neexistujícím provázání se vrací prázdné pole.

V opačném případě (relace *N:1*) použitím příkazu `$zakazka->linka` získáme jediný záznam, se kterým můžeme dále pracovat.

Relace M:N

Pro aplikování takové relace se používají metody `belongToMany()` a `hasMany()`. Zde se pro propojení jednotlivých záznamů použije funkce `attach()`, ke které můžeme přidat tzv. pivot atributy. V tomto případě získáváme přístup k záznamům z tabulky vyjadřujícím vztah mezi provazovanými záznamy. Pro zrušení propojení se používá metoda `detach()`, která odstraní z databáze všechny provázané záznamy ze vztahové tabulky.

5.2.2 Laravel Controller

Pomocí Controlleru dokážeme provádět všechny důležité operace se záznamy v databázi pomocí PHP. Pro každou tabulku můžeme vytvořit příslušný Controller, který implementuje veškeré operace s danou tabulkou a jejími záznamy. Zde lze také jednoduše shromáždit data pro nějaké statistiky a např. vytvářet předlohy pro grafy, které se později vykreslují pomocí JavaScriptu. Ve výchozím nastavení veškeré controllery nalezneme ve složce *app/Http/Controllers* daného projektu.

5.2.3 Směrování - Laravel Routing

Laravel obsahuje vlastní zpracování směrování. Pomocí metod GET, POST, PUT (PATCH) a DELETE dokáže automaticky přesměrovat a případně provést požadované operace. V pří-

padě základních požadavků není potřeba ručně směrovat na stránky, na které se chceme dostat, stačí přenechat směrování Laravelu a pouze určit, který z Controllerů se má o požadavek postarat. Směrování lze upravovat v souboru *web.php*, který je defaultně umístěn ve složce *routes* daného projektu.

Metoda GET

Pro tuhle metodu se v základním požadavku přesměruje na provádění metody `index()` v přiřazeném Controlleru. V případě, kdy chceme např. upravovat nějaký záznam, přidáme parametr (primární klíč záznamu) a budeme pomocí metody GET přesměrováni k provádění funkce `edit()`. Pro zobrazení informací o některém záznamu se směruje na metodu `show()`. Controller poté vyvolá na funkci, která vykreslí výsledek uživateli v prohlížeči.

Metoda POST

Pokud potřebujeme provést nějakou operaci, jako např. uložení nového záznamu do databáze, předáme Controlleru pomocí formuláře požadavek. Laravel automaticky přesměruje na provádění metody `store()`, ve které vytvoříme nový záznam v databázi a přiřadíme hodnoty z požadavku jednotlivým atributům.

Metoda PUT

V případě použití metody `PUT` nebo `PATCH` se používá metoda `update()`, ve které se vyhledá záznam v databázi a následně se upraví atributy podle požadavku. Nakonec se záznam uloží a tím se aktualizuje.

Metoda DELETE

Jestliže se použije metoda `DELETE`, Lavavel začne provádět operace v metodě `destroy()`, ve které se pokusí odstranit záznam z databáze, pokud je to možné. Někdy již nemůžeme odstranit záznam, např. pokud je již propojen s jinými důležitými záznamy v databázi, které již nelze odstranit.

Vlastní funkce

Pokud chceme použít vlastní funkce, framework již automaticky směrování neprovádí. Nyní musíme explicitně vytvořit jednoduchý příkaz, který přesměrování zařídí. V takovém případě lze použít jakoukoli z výše zmíněných metod.

5.3 Tvorba stránek

Pro jednoduché vytváření stránek v HTML využívám Laravel Blade Templates. V souborech s koncovkou *.blade.php* můžeme jednoduše vytvářet stránky pomocí příkazů HTML, ale také používat jednoduchý kód v PHP. Každý soubor typu Blade se přepracuje do kódu PHP a je uložen v mezipaměti. Pokud není modifikován, není potřeba jej znovu kompilovat. Proto je načítání nemodifikovaných stránek prováděno s minimální režií. Všechny stránky jsou v režii Laravel View.

Pro vytváření formulářů existuje nástroj Form, který je součástí Laravelu. Formuláře lze jednodušeji vytvářet a pracovat s nimi.

Pro používání složitějších selectů, ve kterých je potřeba vyhledávat pro zjednodušení výběru nebo třeba zvolit více hodnot, jsem využil nástroj Select2[4], který pomocí JQuery zařídí intuitivní používání tohoto elementu. Pomocí CSS pak upravuje jejich vzhled.

5.3.1 Design stránek

Pro jednotný vzhled webových stránek jsem použil části společné pro každou stránku, hlavním účelem však byla redukce stále se opakujícího se kódu.

Jak již bylo výše zmíněno, pro úpravu vzhledu byl využit nástroj Bootstrap, který pomocí CSS ztvárnil podobu většiny elementů HTML v systému. Bootstrap velmi zjednodušil práci s úpravami designu stránek.

5.4 Řešení problémů

Při implementaci informačního systému jsem narazil na několik zajímavých problémů, které bylo nutné vyřešit. V následující podkapitole jsou problémy popsány a také řešeny.

5.4.1 Speciální vazba generalizace

Tuto speciální vazbu lze vyřešit vícero způsoby. Lze například vytvořit nové tabulky databáze, které vztah vytvoří. Pak je potřeba pomocí některého z atributů v tabulce rozhodnout, jaká z vytvořených tabulek bude použita.

Problém jsem vyřešil sloučením atributů všech figurujících tabulek do tabulky jedné. Jakmile chce uživatel vytvořit nebo upravit záznam takové tabulky, pomocí JavaScriptu se skryjí právě ty části, které nelze použít v kombinaci s rozhodujícím atributem. Taktéž přímo v PHP je nutné ošetřit, aby se jednotlivé záznamy nepřekrývaly.

5.4.2 Grafy

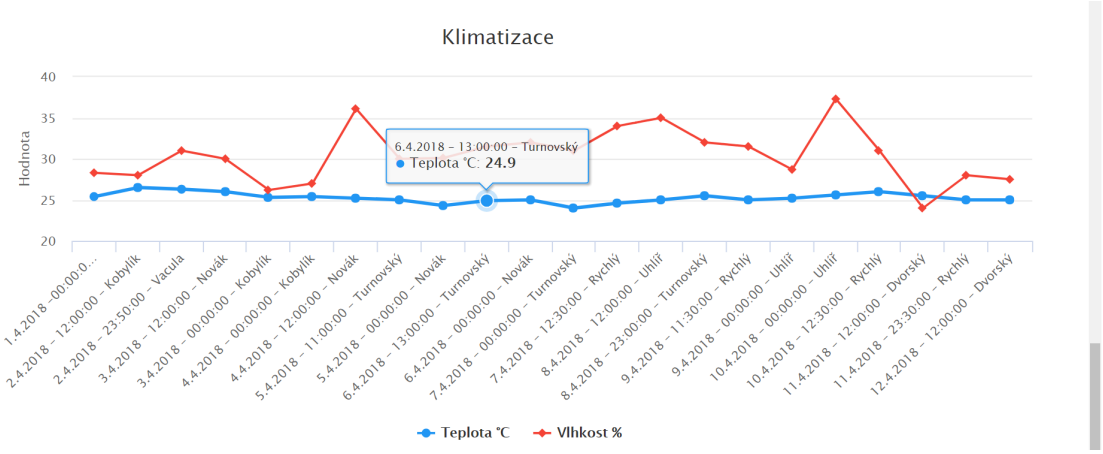
Pro jednoduché zobrazení statistik je nejlepší využít grafů. Jelikož aplikace poběží na offline serveru (nemá přístup k internetu), bylo nutné použít takové nástroje, které nepoužívají online nástroje jako např. Google API. Proto jsem využil soubory JavaScriptu nástroje ConsoleTVs Charts[5]. Při použití grafů HighCharts, které jsou součástí nástroje, lze také s grafy po vykreslení pracovat (např. zobrazovat jen některé z vykreslených částí).

Obrázek 5.1: Graf zobrazující statistiku o plnění

[4]Dostupný na https://select2.org/
[5]Dostupné na https://github.com/ConsoleTVs/Charts

Prozatím bylo požadováno, aby v systému byly jednoduše zobrazeny statistiky o provedené práci zaměstnanců. V grafu 5.1 je pak zobrazeno, na kolik procent směny plnily plán.

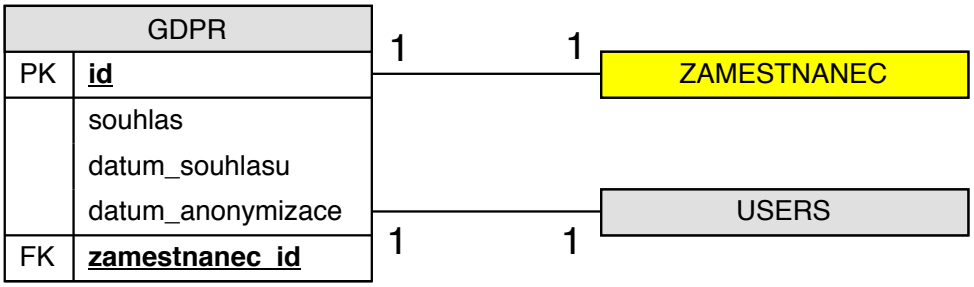
Grafem 5.2 jsou pak zobrazeny i záznamy o klimatizaci.



Obrázek 5.2: Graf zobrazující záznamy o klimatizaci

5.4.3 Vytváření uživatelů a GDPR

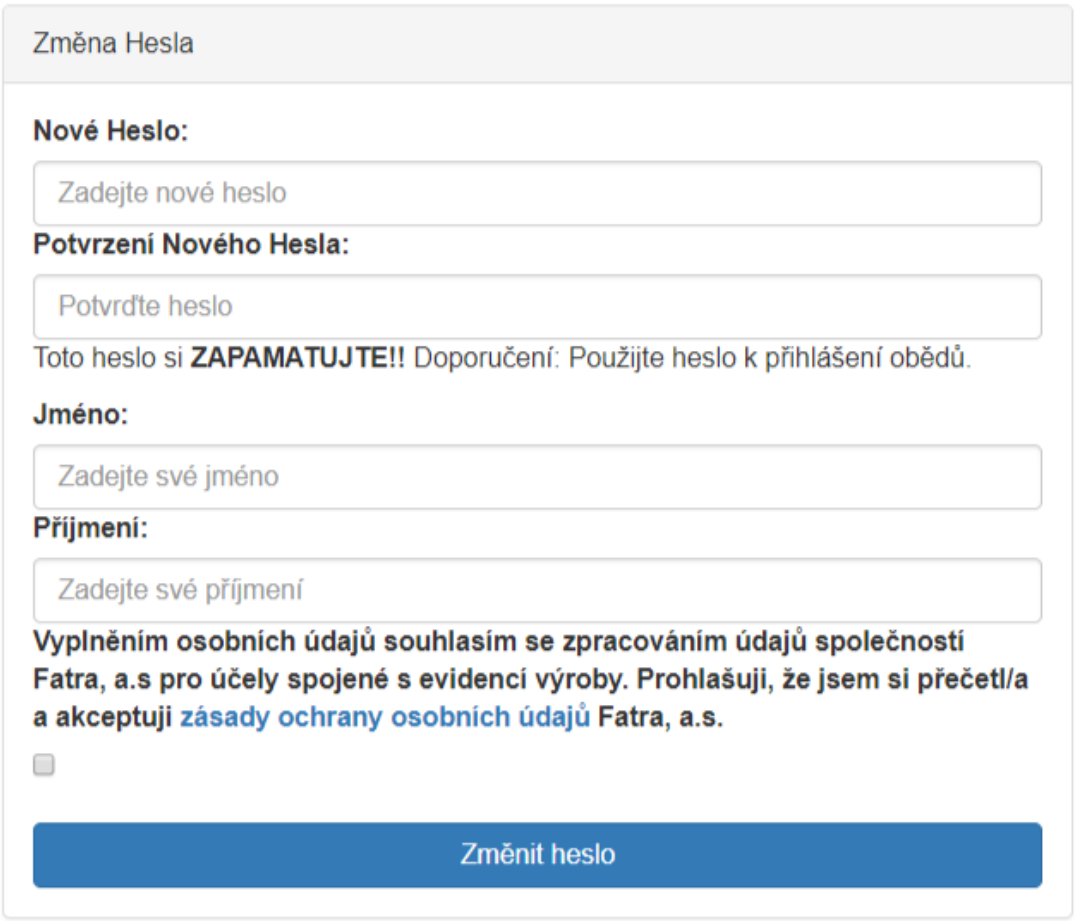
Zadáním uživatelského jména a vybráním role vytvoříme uživatele. Pokud je vybrána role pracovníka, je k němu připojen nově vytvořený nebo neaktivní zaměstnanec. Zaměstnanec má nyní své atributy nastaveny podle uživatelského jména.



Obrázek 5.3: Zjednodušené schéma vztahů pro GDPR

Od 25. 5. 2018 vstupuje účinnost obecné nařízení o ochraně osobních údajů neboli General Data Protection Regulation, které přináší dosud největší revoluci v ochraně osobních údajů pro celou EU[14]. V informačním systému je potřeba udržovat informace o souhlasu o ochraně osobních údajů.

Při prvním přihlášení systém uživatele vyzve ke změně hesla, zadání svých osobních údajů a potvrzení souhlasu s ochranou osobních údajů (viz obr. 5.4). Vytvoří se tak záznam v tabulce *GDPR* (viz obr. 5.3) s potvrzeným souhlasem od uživatele a datem souhlasu.



Obrázek 5.4: Ukázka formuláře prvního přihlášení

Pokud by již uživatel nesouhlasil s tím, že budou jeho osobní údaje v systému vedeny poté, co již nebude jeho uživatelem, lze jeho osobní údaje anonymizovat, přepsat je uživatelským jménem. Tímto se data v systému nebudou moci spojovat s danou osobou. Také o této akci je potřeba uchovat datum provedení.

5.4.4 Vytváření a upravování zakázek

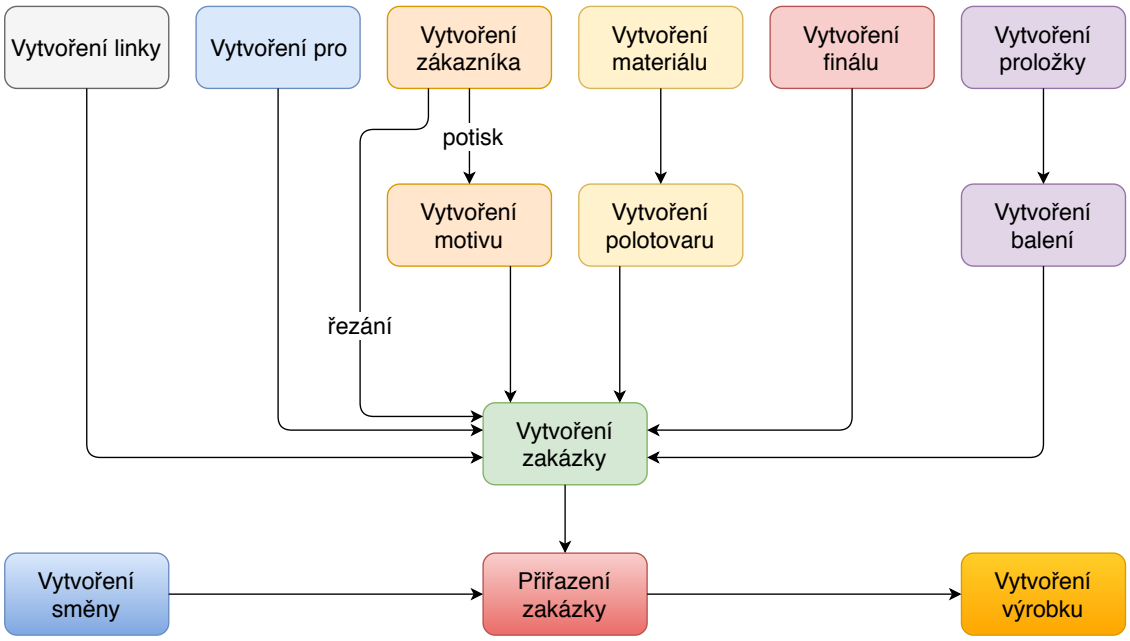
Vytváření zakázek byla z počátku problémová část, protože obsahuje největší množství atributů v tabulce. Pro snadné vytváření jsem použit JQuery, pomocí kterého je z počátku vše skryto. Po zadání povinných polí se postupně další elementy formuláře odkrývají.

Jelikož má zakázka také více cizích klíčů, je potřeba během vytváření vyhledávat nějaké záznamy v databázi. Aby se práce urychlila, je používáno ajaxové volání, které postupně načítá veškeré záznamy do selectů. Pak již lze pouze volit existující záznamy. Vždy je potřeba potvrdit vybraný záznam.

Také lze přidávat poznámky, které lze vybrat a upravit, nebo napsat nové. Vše je zpracováno pomocí JQuery.

Dále se automaticky počítá množství nebo počet rolí k výrobě. Pokud zadáme počet palet, vypočítá se automaticky obojí.

Před vytvořením zakázky je nutné mít vytvořeny všechny její části. Postup takového vytvoření je ukázán na obr. 5.5.



Obrázek 5.5: Postup při vytvoření první zakázky a postup k vytvoření výrobku

5.4.5 Změna pořadí zakázek

Důležitým požadavkem byla jednoduchá změna pořadí jednotlivých zakázek prováděných na určité lince. Tento problém jsem vyřešil použitím JQuery User Interface[6]. Použitím funkce `sortable()` z této knihovny se tabulka prováděných zakázek promění v editovatelnou v podobě přesunů jednotlivých řádků. Po provedení přesunu se provede volání pomocí AJAXu s indexem počátku a konce. Funkce poté přepíše všechny zakázky, které změna pořadí ovlivní. Po dokončení se stránka zaktualizuje.

5.4.6 Vytváření a úpravy směny

Uživatel s rolí pracovníka může vytvářet směnu, která musí mít minimálně 1 zaměstnance v roli tiskaře nebo řezáče (podle linky, výběr omezen pomocí JQuery). Při vytvoření směny se pro všechny zaměstnance směny automaticky vytváří záznam do tabulky *Týdeníček*. Tento záznam uchovává informace o tom, kdy byl zaměstnanec na směně a jak plnil daný plán. Úpravou směny se upraví i přiřazené týdeníčky. Po vytvoření směny lze přiřadit směně zakázku na dané lince a poté lze provádět výrobu přidáváním nových výrobků (viz obr. 5.5). Směnu lze odstranit, pokud s ní nebyly provedeny žádné změny v systému kromě vytvoření týdeníčků.

[6]Dostupné zdarma na http://jqueryui.com/download/

5.4.7 Vytváření týdeníčků

Ke konci měsíce je potřeba vytvořit předběžně záznamy o docházce. Lze zadat počet hodin, přidělit roli zaměstnance na směně a přiřadit jej na určitou linku. V případě, kdy bude mít zaměstnanec např. volno, zapíše počet hodin, ale nemá možnost přiřadit žádnou linku.

Jakmile je vytvořena směna v den, pro který již existuje týdeníček, je týdeníček aktualizován.

5.4.8 Vytváření dalších operací

Další operace jsou vždy vázány k aktuální směně uživatele, protože se může stát, že uživatel zapomene něco provést. Může to provést příště, avšak pouze do doby, dokud nevytvoří novou směnu.

5.4.9 Sklad barev a ředidel

Pro evidenci barev a ředidel není potřeba vědět kdo a kdy přesně přijmul, odebral nebo vrátil barvu či ředidlo na sklad. Důležitými údaji jsou pouze typ operace a hmotnost. Při provedení jakékoliv operace se stav barev ihned aktualizuje. Systém zobrazuje pro každý den v měsíci sumu hmotností podle provedených typů operací za ranní a noční směny.

Kapitola 6

Testovaní

Testování probíhalo ve dvou fázích. V první fázi jsem testoval systém sám, kdy jsem vytvářel mnoho možných vstupů, abych systém ošetřil proti široké škále chyb. Chyby, které by negativně ovlivnily databázi, byly odstraňovány během vytváření aplikace.

Problémy, na které se bylo nutno zaměřit:

- vytváření směn - automatické vytvoření týdeníčků,
- vytváření zakázek - neočekávané chyby ajaxového volání,
- vytváření a mazání výrobků - změna aktuálního plnění,
- editace směn - automatická editace týdeníčků,
- odstraňování záznamů z databáze.

Vytváření některých záznamů je spojeno s automatickým vytvářením dalších záznamů. Automatizace vytváření těchto záznamů z počátku nebyla dokonalá, v době implementace aplikace však bylo vše doladěno. Taktéž v případě editace záznamů vznikaly problémy s těmito automatickými záznamy.

Bylo nutné doladit vzorec plnění plánu jednotlivých směn. V dřívějším systému nebylo vyřešeno počítání plnění v případě vícero zakázek. Směna tudíž musela dopočítávat, kolik je potřeba ještě vyrobit, aby splnila plán.

V případě odstraňování záznamů nastaly chyby s porušováním integrity. Během odstraňování byly některé části odstraněny a poté zpracování zastavila kontrola cizích klíčů. Konzistence tak byla porušena a části, které jsme chtěly odstranit, zůstaly v databázi.

6.1 Různé operační systémy

I když je aplikace vyvíjena pro systém Microsoft Windows, testoval jsem jej i na operačním systému Linux. Ve Windows žádné problémy nenastaly, u Linuxu vznikl problém při vytváření operací s aniloxy. Využití elementů select nástroje Select2 s povolením více možností u Windows vytvořilo pole všech elementů, i když nebyly změněny. U Linuxu bylo vytvořeno pole pouze editovaných elementů. Bylo potřeba přidat implicitní hodnoty všech elementů select, aby se počet elementů srovnal.

6.2 Testování ve firmě Fatra, a.s.

Druhou fází testování bylo přímo ve výrobě. Jelikož se systém v prvních fázích velice líbil, rozšířil jsem systém o linku typu řezačka. Rozšíření nebylo složité, protože se pouze přidaly atributy do tabulky *Zakázka* a *Výrobek*. Dále bylo potřeba poupravit výchozí zobrazení menu, jelikož některé části jsou potřeba u směn na potiskovací lince. Nakonec bylo upraveno zobrazení jednotlivých zakázek a výrobků. Testování poté probíhalo skoro 2 týdny, jelikož se většina částí používá stále, byla testovací data pro tuto práci dostatečná.

6.2.1 Výsledek testování

Během testování se odhalilo několik velmi malých nedostatků (vzhled, chyby funkcí implementovaných v JavaScriptu, načtení nepožadovaných záznamů), které byly ošetřeny během pár minut. Jako odezvu na testování jsem využil Google Forms[1] s jednoduchými otázkami, které jsou popsány v příloze B.

Výsledky testování byly uspokojivé, většina pracovníků by systém doporučila k dalšímu vývoji. Ti, kteří by systém nedoporučovali, jsou zvyklí na zaběhnutý režim, který nechtějí obměňovat.

Během testování několik pracovníků napadlo další rozšíření systému, které je popsáno v podkapitole 7.2.

[1] Výsledky dostupné na https://goo.gl/STUje3

Kapitola 7

Další vývoj systému

Jelikož se během testování systém víceméně zalíbil, je pravděpodobné, že se bude systém dále rozšiřovat. V následujících podkapitolách se podíváme, jaké jsou možnosti dalšího vývoje tohoto informačního systému (dále IS).

7.1 Optimalizace pro další zařízení

V současné době IS běží pouze na stolních počítačích, to by se však dalo pro vedení upravit tak, aby mohli výrobu sledovat takřka odkudkoli ve firmě (v případě povolení přístupu online k serveru i zvenčí). Bylo by potřeba optimalizovat evidenci pro mobilní zařízení.

Také by byla možnost použití tabletu ve skladu barev a ředidel, zatím to však nebylo požadováno.

7.2 Plánování směn

Během testování si několik pracovníků přálo přidat do systému plánování směn. Po konzultaci jsem zjistil, že by chtěli vědět, která zakázka se pojede na nějaké z budoucích směn.

Takové plánování by však muselo počítat s určitými časovými prodlevami, které vznikají během změny zakázky. V rámci několika měsíců sledování procesu změn zakázek by se dal získat určitý koeficient, kterým by se získávala prodleva mezi měněnými zakázkami.

Po získání koeficientu by bylo možné alespoň přibližně určit, co se na jaké směně bude provádět.

7.3 Rozšíření o další linky

Oddělení PPF není jediným oddělením spojeným s touto výrobou, všechna propojená oddělení využívají systém MS Office Excel. V ER diagramu A.1 lze vidět tabulku polotovar. Polotovary jsou vyráběny na lince oddělení PPL (Paropropustné lamináty). Pokud by se systém rozšiřoval, opět by bylo potřeba evidovat materiál, recyklace odpadu a jednotlivé výrobky - polotovary. Takové výrobky by poté byly evidovány v IS. Evidence by se pak nevztahovala pouze k určitému typu polotovaru, ale k polotovaru samotnému.

Výrobky by mohly být zpracovány dalšími dvěma výrobními linkami, které by také byly přidány do IS.

Existuje linka, která dále zpracovává výrobky, jež jsou nyní vytvářeny v IS. Zde by musela být přidána evidence materiálu a lepidel.

7.4 Spolupráce s firemní databází

V případě, že by vše fungovalo přesně jak má a nenastávaly by příliš velké odchylky, které by byly nutno opravovat, by mohl IS spolupracovat s informačním systémem SAP. V takovém případě by mohly být zakázky vytvářeny poloautomaticky, jelikož informace jsou převzaty s této databáze. Taktéž inventury na konci měsíce by probíhaly automaticky a vedení oddělení by ušetřilo velmi mnoho času a práce.

7.5 Spolupráce se mzdovým systémem

Pokud se osvědčí práce s týdeníčky, bude mít IS možnost spolupracovat se mzdovým informačním systémem. Takové rozšíření by však bylo časově náročné, jelikož jde o velmi důležitou práci, při které může být mnoho zaměstnanců a také samotná firma obrána o peníze, není zde možnost dovolit si jakékoliv chyby. Taková automatizace by však vedoucím ušetřila nejméně 24 hodin práce měsíčně.

7.6 Spolupráce s evidencí systému MES

V blízké době bude využita evidence výrobků pomocí čárových kódů v systému MES. Taková evidence by se dala také rozšířit do IS. Propojením systémů by se opět snížil počet prováděných akcí k evidenci těchto výrobků.

Kapitola 8

Závěr

Jak již bylo popsáno v úvodu, cílem práce bylo vytvořit informační systém pro sledování výroby. Nejprve bylo potřeba zanalyzovat veškeré procesy spojené s evidencí výroby, abych mohl co nejpřesněji definovat, jak má jádro systém vypadat. S touto analýzou mi pomohl klient a jeho pracovníci, kteří mi poskytli veškeré potřebné informace. Po analýze byly vytvořeny diagramy případů užití, které určily, kdo a jak se systémem bude moci pracovat. Pro vytvoření databáze byly použity ER diagramy, které určují, jaké jsou vztahy mezi jednotlivými entitami.

Po získání všech náležitostí byla implementována aplikace s použitím nástroje Laravel. Nakonec musel být systém otestován, vše probíhalo ve dvou fázích. V první jsem systém testoval sám, vkládal jsem vstupy, odstraňoval záznamy apod. Odhalil jsem tak více než 75 % chyb ještě před testováním ve firmě. V druhé fázi již bylo provedeno testování ve firmě, kde systém běžel na dvou linkách. Na konci všech těchto procesů byl vytvořen jednoduchý dotazník s cílem získat odezvu od pracovníků na vzniklý systém.

Pro tento informační systém mám vymyšleno několik vylepšení, která budou v blízké době prováděna. Jedním z nich je například rozšíření systému na další oddělení ve firmě, které je úzce spjato s oddělením, pro které byl systém vyvíjen. Navíc s pomocí nástroje Bootstrap nebude problém systém optimalizovat pro mobilní zařízení a tablety, takže bude možné systém využívat prakticky odkudkoli.

Při práci na tomto projektu jsem získal bohaté zkušenosti při komunikaci s klientem. Jelikož jsem nikdy dříve nevytvářel aplikaci pomocí nástroje Laravel, bylo nutné se jej naučit používat. Použití tohoto nástroje mi ušetřilo mnoho práce, proto jsem velice spokojen, že jsem si vybral právě Laravel. Největší zkušeností však bylo vytvoření celého informačního systému, včetně jeho testování a údržby.

Literatura

[1] *About Bootstrap*. [Online; navštíveno 12.03.2018].
URL https://getbootstrap.com/docs/4.0/about/overview/

[2] Chaffer, J.; Swedberg, K.; Baše, K.; aj.: *Mistrovství v JQuery: kompletní průvodce vývojáře*. Computer Press, 2013, ISBN 978-80-251-4103-8.

[3] Darie, C.; Skřivánek, R.: *AJAX a PHP - tvoříme interaktivní webové aplikace profesionálně*. Zoner Press, 2006, ISBN 80-86815-47-1.

[4] Fatra: *Profil společnosti | Fatra. Plasty pro život*. [Online; navštíveno 17.01.2018].
URL http://www.fatra.cz/o-nas/profil-spolecnosti/

[5] Flanagan, D.: *JavaScript: kompletní průvodce. 2. aktualiz. vyd.* Computer Press, 2002, ISBN 80-7226-626-8.

[6] Friends, A.: *XAMPP Installers and Downloads for Apache Friends*. [Online; navštíveno 17.01.2018].
URL https://www.apachefriends.org

[7] Janovský, D.: *Úvod do CSS*. [Online; navštíveno 17.01.2018].
URL https://www.jakpsatweb.cz/css/css-uvod.html

[8] Ľuboslav Lacko: *PHP a MySQL: hotová řešení*. CP Books, 2005, ISBN 80-251-0397-8.

[9] Naramore, E.; Kiszka, B.: *PHP5, MySQL, Apache: vytváříme webové aplikace*. Computer Press, 2006, ISBN 80-251-1073-7.

[10] Rejnková, P.: *Příklady použití diagramů UML 2.0 . Příklady použití diagramů UML 2.0*. [Online; navštíveno 17.01.2018].
URL http://uml.czweb.org/pripad_uziti.htm

[11] Stejskal, J.: *Vytváříme WWW stránky pomocí HTML, CSS, a JavaScriptu*. Computer Press, 2004, ISBN 80-251-0167-3.

[12] Welling, L.; Thompsonová, L.: *PHP a MySQL: rozvoj webových aplikací*. SoftPress, 2003, ISBN 80-86497-60-7.

[13] Wikipedia: *Laravel*. [Online; navštíveno 22.04.2018].
URL https://cs.wikipedia.org/wiki/Laravel

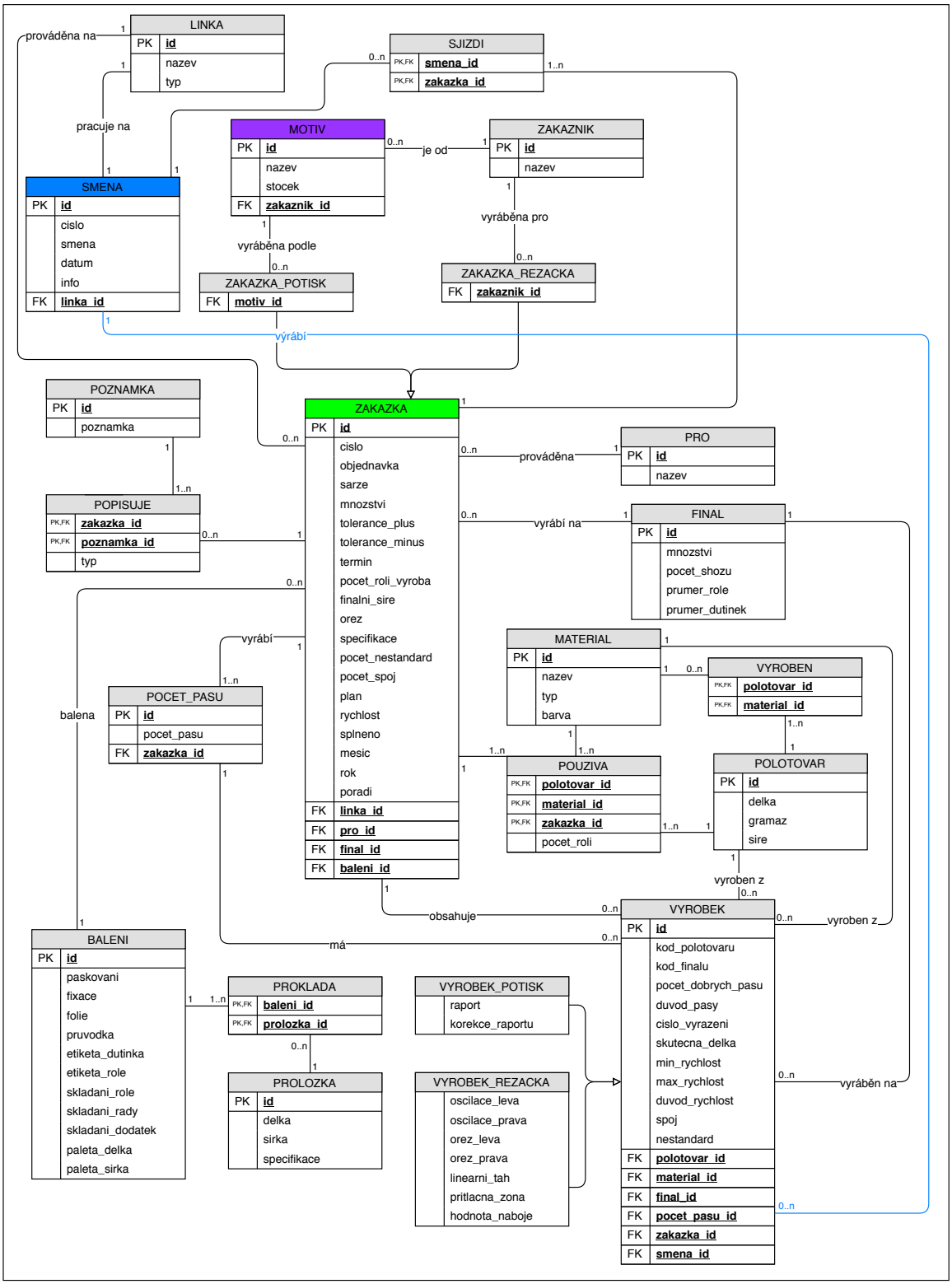
[14] Škorničková, M. E.: *GDPR | Obecné nařízení o ochraně osobních údajů — prakticky*. [Online; navštíveno 28.04.2018].
URL https://www.gdpr.cz/

Příloha A

Obrázky

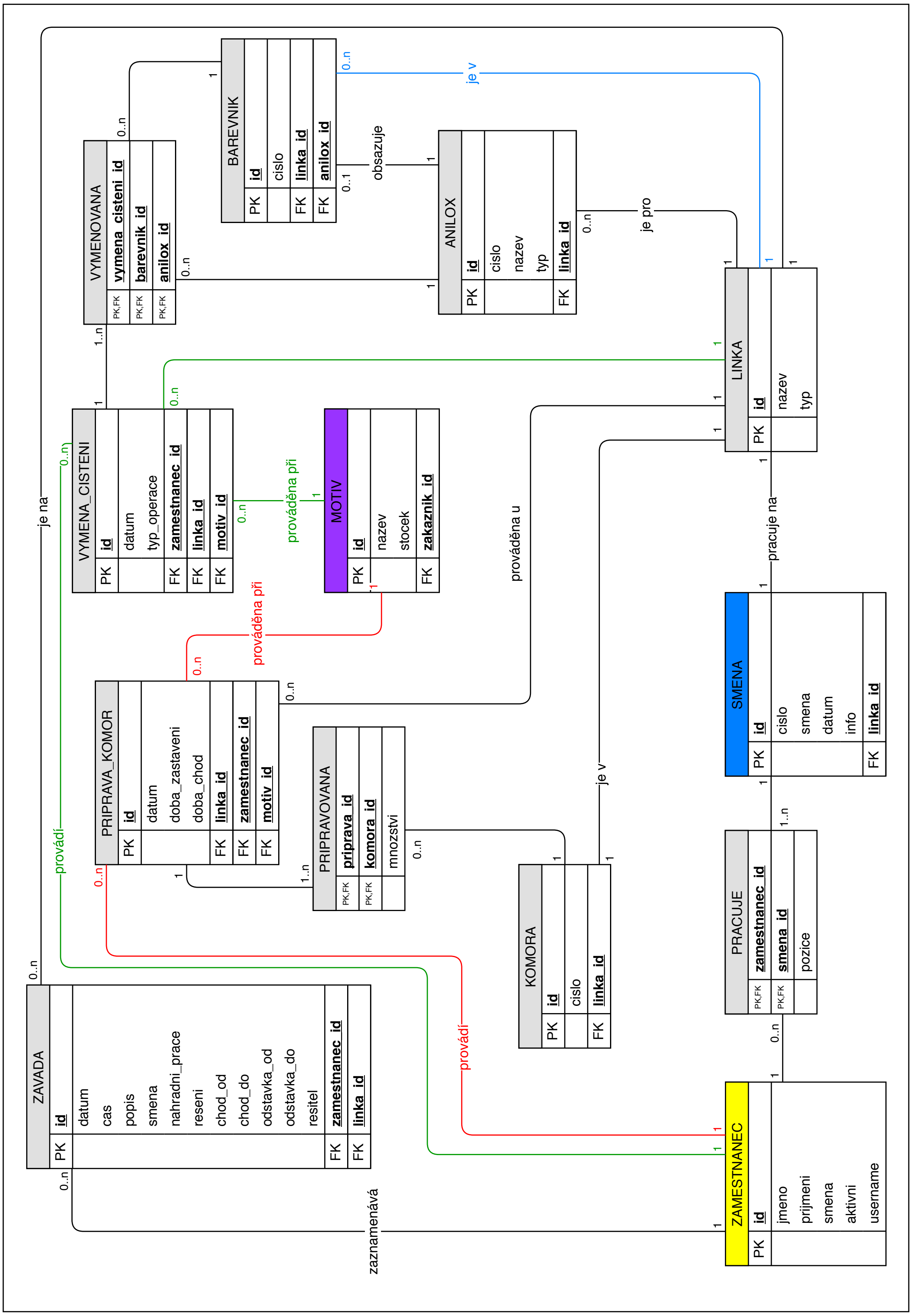
A.1 Seznam obrázků

A.1.1 ER diagram – část Zakázka



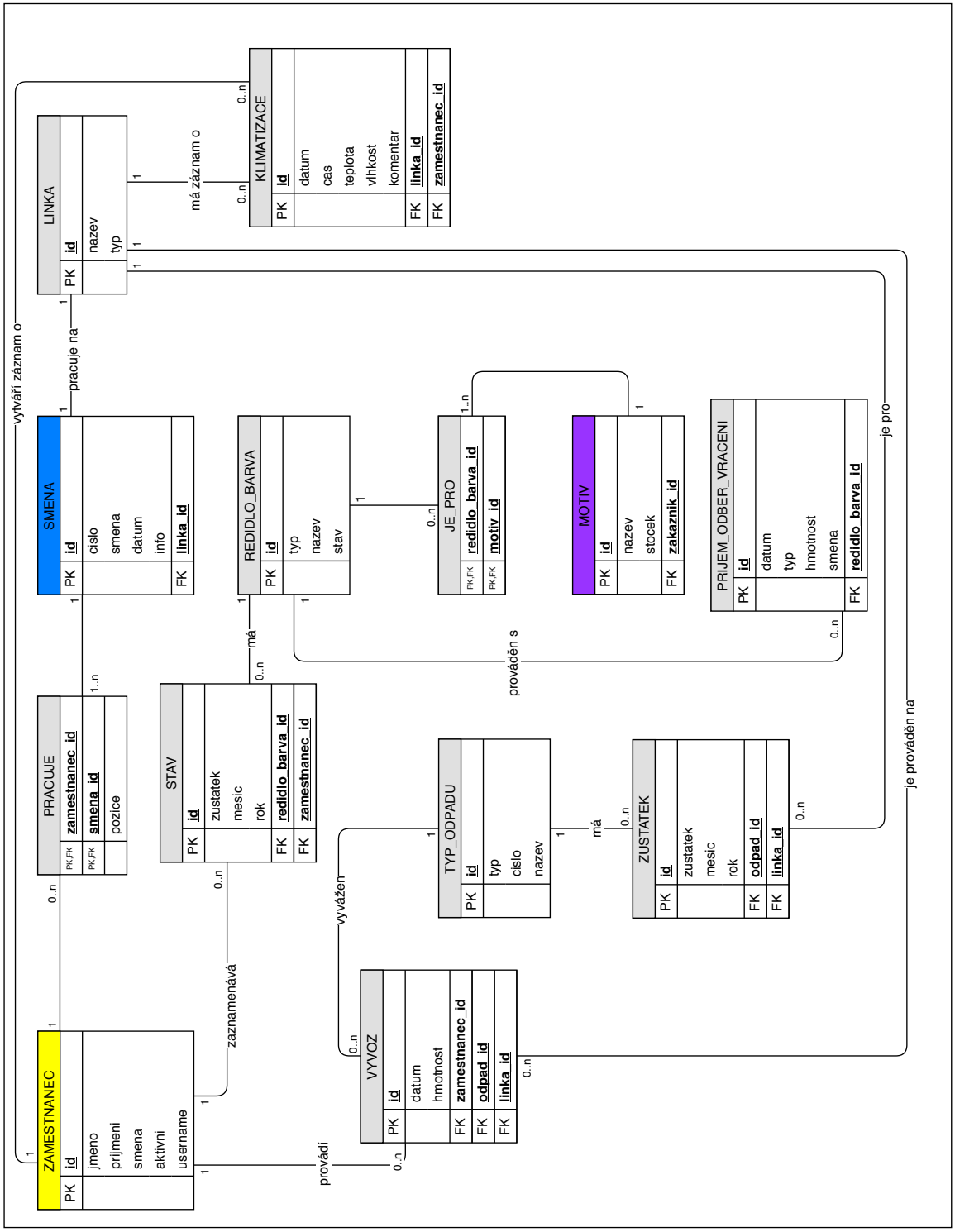
Obrázek A.1: ER diagram – část Zakázka

A.1.2 ER diagram – část Operace



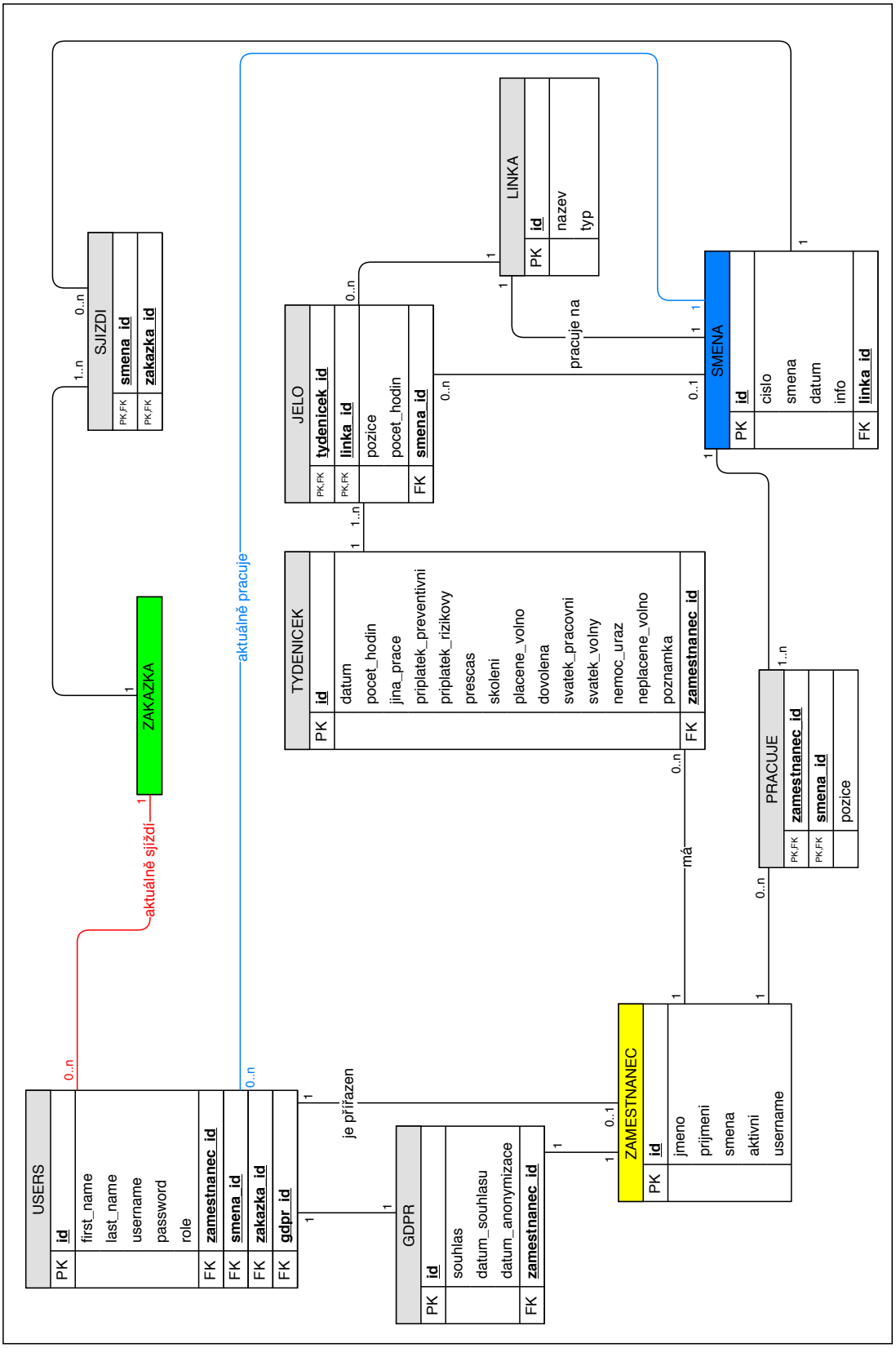
Obrázek A.2: ER diagram – část Operace

A.1.3 ER diagram – část Vývozy



Obrázek A.3: ER diagram – část Vývozy

A.1.4 ER diagram – část Týdeníček



Obrázek A.4: ER diagram – část Týdeníček

Příloha B

Dotazník

1. Ohodnoťte na stupnici, jak moc se Vám systém líbí.
2. Ohodnoťte na stupnici, jak moc souhlasíte s nahrazením současného systému.
3. Jak se Vám líbí vzhled informačního systému?
4. Jak obtížná je orientace v informačním systému?
5. Jste (vedení nebo pracovník)?
   (a) Jak pro Vás bylo složité vytvořit novou zakázku?
   (b) Jak pro Vás bylo složité vytvořit nový výrobek?
6. Stručně popište, co bych měl udělat pro zlepšení systému.
7. Doporučíte systém k dalšímu vývoji?
   (a) Popište proč.

Příloha C

Obsah přiloženého CD

- **./doc/projekt.pdf** - text bakalářské práce
- **./doc/src** - adresář se zdrojovými kódy textu bakalářské práce
- **./src** - adresář se zdrojovými kódy informačního systému
- **./src/readme.txt** - návod na instalaci