
**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 FIRMY ZABÝVAJÍCÍ SE POVRCHOVÝMI ÚPRAVAMI

**INFORMATION SYSTEM OF A COMPANY FOCUSED ON SURFACE TREATMENT**

**BAKALÁŘSKÁ PRÁCE**
**BACHELOR'S THESIS**


**AUTOR PRÁCE** **MATÚŠ VIŠČOR**
**AUTHOR**

**VEDOUCÍ PRÁCE** **Ing. VLADIMÍR BARTÍK, Ph.D.**
**SUPERVISOR**



**BRNO 2021**

**Vysoké učení technické v Brně**
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

# Zadání bakalářské práce


Student: **Viščor Matúš**
Program: Informační technologie
Název: **Informační systém firmy zabývající se povrchovými úpravami**
**Information System of a Company Focused on Surface Treatment**
Kategorie: Informační systémy
Zadání:

1. Seznamte se s prostředím firmy zabývající se povrchovými úpravami (lakováním) a možnými vývojovými prostředky pro tvorbu informačních systémů (webové/desktopové).
2. Analyzujte požadavky na informační systém pro správu této firmy zahrnující správu zakázek a skladových zásob, který mj. bude plánovat činnost tak, aby docházelo k minimálnímu zdržení, nabízet statistiky o spotřebě barvy od různých výrobců za účelem úspory a různé možnosti exportu dat.
3. Navrhněte informační systém dle požadavků, použijte přitom vhodné modelovací techniky jazyka UML.
4. Implementujte navržený informační systém a proveďte jeho testování na vhodném vzorku dat.
5. Zhodnoťte dosažené výsledky a další možnosti pokračování tohoto projektu.

Literatura:

- Žára, O.: JavaScript - Programátorské techniky a webové technologie, Computer Press, 2015. ISBN: 978-80-251-4573-9.
- Welling, L., Thomson, L.: PHP a MySQL: Kompletní průvodce vývojáře. CPress, 2017.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Bartík Vladimír, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 22. října 2020

# Abstrakt

Táto práca rieši analýzu požiadavok firmy zaoberajúcej sa povrchovými úpravami a následne návrh a implementáciu informačného systému. Kladie sa dôraz na správu zakázok, ktorým sa bude plánovať činnosť tak, aby dochádzalo k minimálnemu zdržiavaniu výroby vo firme. Systém ponúka aj správu skladových zásob rôznych farieb od rôznych výrobcov a ponúka štatistiky o ich spotrebe za účelom analýzy. Na riešenie zvoleného problému boli použité UML diagramy pre návrh systému a framework Laravel spolu s javascriptovímy knihovnami pre jeho následnú implementáciu.



# Abstract

This work solves the analysis of the requirements of the company engaged in surface modifications and subsequently a proposal and implementation of the information system. It is emphasized to manage contracts to plan the activity to commute a minimum resident of production in the company. The system also offers management stocks of various colors from different manufacturers and offers statistics on their consumption for analysis. UML diagrams for your design and Framework Laravel together with JavaScript libraries for its subsequent implementation were used to address the selected problem.



# Kľúčové slová

Informačný systém, práškove lakovanie, povrchové úpravy, firma, webové technológie, PHP, Laravel, Framework, MySQL, jQuery, Datatables, HTML, CSS, Javascript

# Keywords

Information system, Surface treatment, powder coating, company, web technologie, PHP, Laravel, Framework, MySQL, jQuery, Datatables, HTML, CSS, Javascript


# Citácia

VIŠČOR, Matúš. *Informační systém firmy zabývající se povrchovými úpravami*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vladimír Bartík, Ph.D.

# Informační systém firmy zabývající se povrchovými úpravami

## Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing.Vladimíra Bartíka, Ph.D. Ďalšie informácie mi poskytol pán Ing. Zdenko Brtáň. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Matúš Viščor
11. mája 2021


## Poďakovanie

Rád by som vyjadril poďakovanie vedúcemu tejto bakalárskej práce, pánovi Ing. Vladimírovi Bartíkovi, Ph.D, za poskytnuté konzultácie a podporu. Ďalej by som rád poďakoval Ing. Zdenkovi Brtáňovi, ktorý je majiteľom spoločnosti zaoberajúcej sa povrchovými úpravami kovov STILMAN GROUP s.r.o, za jeho spoluprácu pri špecifikácií požiadavok a za prínos nových nápadov, ktoré posúvali vývoj informačného systému vpred.

# Obsah

# Kapitola 1

# Úvod

V akomkoľvek podniku či firme je proces, alebo súbor činností, ktoré treba vykonávať za účelom dosiahnuť požadovný cieľ. Každá podnikateľská činnosť má svoje procesy a postupy. Firmy dosahujú vyššiu efektivitu prostredníctvom optimalizovania týchto procesov. Jedným z nástrojov, ako procesy optimalizovať je ich čiastočná alebo úplná automatizácia a riadenie pomocou softvérových riešení. Takýmto softvérovým riešením je určite aj informačný systém. Podnikový informačný systém sa snaží eliminovať nestabilitu pracovných procesov firmy a udržuje tak poriadok pri vykonávaní každodenných časovo náročných úloh, čo napomáha k zmenšovaniu výskytu ľudských chýb. Podnikový informačný systém sa vždy odvíja od špecifických potrieb a charakteru podnikania. O naplnenie týchto špecifických potrieb sa bude snažiť aj táto bakalárska práca.

Informačný systém, popísaný v tejto práci a zároveň slúžiaci na zlepšenie výrobného procesu bol vytvorený na požiadavku a v spolupráci s podnikom STILMAN GROUP s.r.o, ktorý sa zaoberá povrchovými úpravami kovov. Cieľom tejto bakalárskej práce je navrhnúť a vytvoriť informačný systém vzhľadom na potreby a fungovanie firmy. Pri návrhu systému sa kladie dôraz na správu a efektívne plánovanie jednotlivých zakázok a pomôcť tak zamestnancom firmy vytvoriť, čo najlepší plán výroby. Podobne sa kladie dôraz aj na správu skladových zásob firmy, ktoré predstavujú práškovacie farby doplnené o štatistiky využívania týchto farieb. Štatistiky tak môžu pomôcť k účinnému využívaniu skladových zásob do budúcnosti. Systém poskytuje aj možnosť exportu dát, kvôli potrebe pracovať s dátami aj mimo systému. Systém obsahuje však aj ďalšie zlepšujúce prvky, ktoré urýchlia a zjednodušia fungovanie firmy.

Bakalársku prácu je možné rozdeliť na dve časti. Zatiaľ, čo prvá časť sa skôr venuje firme, jej prostrediu a samotnému procesu výroby. Druhá časť sa venuje už informačnému systému, jeho návrhu, použitým technológiám a implementácii tohoto systému.

Prostrediu firmy sa práca venuje v kapitolách 2 a 3. V kapitole 2 práca popisuje odvetvie, ktorým sa firma zaoberá. Kapitola 3 sa zase zaoberá, súčasným riešením fungovania firmy a načrtuje aj nové riešenie v podobe jednotlivých častí systému vzhľadom na požiadavky tejto firmy.

Druhej časti práce sa venujú kapitoly 4, 5 a 6, ktoré sú už viac praktickejšie. V kapitole 4 je popísaný návrh celého informačného systému. Kapitola 5 popisuje použité webové technológie pri implementácii. Kapitola 6 popisuje jeho implementáciu. Stručne popísané samotné testovanie implementovanej časti projektu sa nachádza v kapitole 7. V kapitole 8 je obsiahnuté zhrnutie celého riešenia tejto bakalárskej práce spolu s možnými rozšireniami.

# Kapitola 2

# Úvod do problematiky práškového lakovania

Pri vývoji podnikového informačného systému, je dobre aspoň sa z ľahká oboznámiť s problematikou, ktorou sa firma zaoberá. Následne požiadavky by mali potom dávať väčší zmysel a bude možné požadované riešenie od zákazníka rozšíriť ešte o kvalitnejšie riešenie.

## 2.1 Princíp

Práškové lakovanie, tiež nazývané ako komaxitovanie, je proces, kde je nanášacia vrstva reprezentovaná voľne tečúcim, suchým práškom. Hlavný rozdiel medzi konvenčnými kvapalnými farbami a práškovými farbami je, že práškové farby nepotrebujú rozpúšťadlá. Vrstva farby sa nanáša na povrch a potom sa vytvrdí za tepla. [21] Najbežnejší spôsob nanášania práškových farieb je proces nazývaný elektrostatické nanášanie striekaním, ktorým sa dosiahne nanášanie vrstvy na kovový podklad. Na vytvorenie práškového povlaku na kovovom podklade je potrebná striekacia pištoľ, ktorá zabezpečí nanášanie elektrostatického náboja na častice prášku. [19] Pištoľ odovzdá prášku kladný elektrický náboj, ktorý sa potom nastrieka na uzemnený predmet, mechanickým alebo pneumatickým striekaním, ku ktorému je vplyvom elektrostatického náboja silne priťahovaný. Výsledná vrstva je tvrdšia, ako pri bežných farbách. [21] Po nanesení práškového povlaku sa kovový predmet zahreje v peci, čo umožní chemickému reagovaniu povlaku a vytvoreniu nového povrchu. [19]

## 2.2 Práškové farby

Práškové farby sa používajú predovšetkým na nátery kovov, ako sú napríklad kovové časti domácich spotrebičov, hliníkových líšt a profilov, bicyklov, rôznych konštrukcií a zvárencov, brány rodinných domov, diskov automobilov a pod. Práškové farby môžu byť určené pre exteriér alebo interiér. Ďalej sa delia podľa odtieňa, typu povrchu a každá farba môže pochádzať od iného výrobcu. [21]

## 2.3 Príprava pred lakovaním

Príprava pred nanášaním práškových farieb spočíva v odmastení, odstránení iných nečistôt, oxidov kovov, pozostatkov po zváraní. Tieto úkony môžu byť vykonávané pomocou

rôznych chemických a mechanických metód. Okrem prípravy povrchu, pri ktorom dochádza k odstráneniu nečistôt môže byť potrebné zvážiť zdrsneniu povrchu, čo priaznivo pôsobí na priľnavosť náteru k povrchu výrobku. [21]

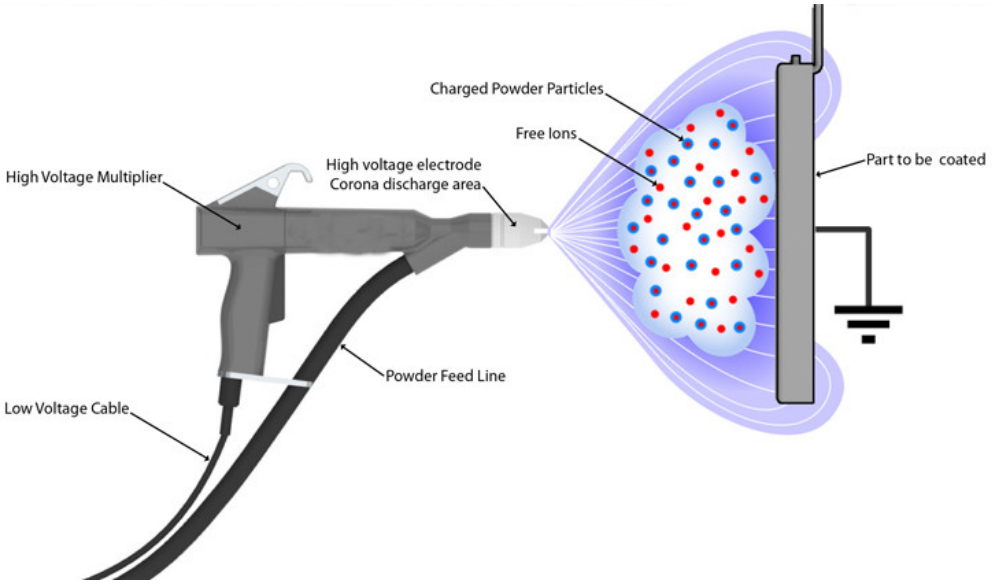
## 2.4 Dôležité poznatky pre informačný systém

Z princípu práškového lakovania je vhodné zdôrazniť, že predmety, ktoré sa lakujú, musia byť zavesené špecifickou technikou v striekacej kabíne a po procese striekania sa nesmie narušiť prášok na kovovom povrchu. Pre návrh informačného systému je dobre si uvedomiť, potrebu zoskupenia predmetov na lakovanie, aby sa na nich jedným ťahom pištoľa nanieslo čo naviac farby.

Jeden z ďalších poznatkov je nanášanie prášku, ktorý vyžaduje bezprašné a čisté prostredie. Pokiaľ toto nie je zabezpečené môže veľmi ľahko dôjsť k znečisteniu práškovej farby ako aj k nasadaniu nečistôt na predtým už vyčistený povrch výrobkov, čo sa v konečnom dôsledku prejaví na vzhľade povrchu. Je to jeden z veľmi častých a ťažko riešiteľných problémov. [21] Vzhľadom na tento problém je potrebné pri zmene farby (keďže znečistenie rovnakou farbou nemá vplyv), prečistenie celej kabíny a prebitia pištoľa inou farbou, čo trvá aj desiatky minút. Je, teda kľúčové poznať pripravené zákazky rovnakej farby a snažiť sa ich zoskupiť za účelom úspory času.

Poznatky o farbách napovedajú, že jedna farba nebude jednoznačné určená len jedným atribútom, ktorý predstavuje odtieň, ale na sklade sa ich môže s jedným odtieňom nachádzať viac.

Vďaka informáciám o príprave pred lakovaním je zrejme, že farba nie vždy dobre drží na lakovanom predmete vplyvom materiálu, prípravy či výrobcom farby. Aby bolo jednoduchšie sa dopátrať k chybe alebo zistiť nové poznatky o procese je vhodné zaznamenávať využitú farbu pri striekaní a porovnávať s inými zákazkami. Zaznamenávanie farby tiež napomáha k lepšej predstave o množstve farby potrebné na nasledujúce obdobie.




Obr. 2.1: Obrázok zobrazuje princíp práškového lakovania (zdroj: [9])

# Kapitola 3

# Požiadavky na informačný systém

Veľmi dôležitým procesom pri vývoji informačného systému je špecifikácia požiadaviek na informačný systém. Je kľúčové zoznámiť sa s prostredím firmy a analyzovať ich súčasné vykonávanie činnosti, spojené s výrobou lakovaných produktov.

Jedna z požiadavok firmy, bola vytvoriť webovú aplikáciu, a to z nasledujúcich dôvodov. Firma by chcela pristupovať do systému z ktoréhokoľvek zariadenia bez nutnosti inštalácie a riešenia otázky obmedzenia hardvérom. Zamestnanci uvažujú využívať mobilné zariadenia hlavne k rýchlemu prístupu k dátam a získavaniu informácií z nich. Nebude príliš časté, aby zamestnanec zadával dáta do systému z mobilného zariadenia. Preto nie je veľmi vhodné investovať čas do vývoja mobilnej aplikácie, avšak prezeranie dát cez mobilné zariadenie je takisto požiadavkou. Z tohoto dôvodu bola zvolená alternatíva responzivného webu.

V tejto kapitole je uvedené, ako lakovacia firma funguje, čo je pre nich dôležité, ako funguje bez informačného systému a návrh nového riešenia vzhľadom na požiadavky firmy. Je potrebné upozorniť, že návrh tohoto systému vychádza z konkrétnych požiadavok firmy, ktorá sa problematike povrchovej úpravy kovov venuje. Požiadavky boli získané pravidelnými stretnutiami s majiteľom firmy a osobnými návštevami výrobnej haly. Informácie v tejto kapitole sú podložené reálnymi skúsenostiami zamestnancami danej firmy.

## 3.1 Zoznámenie s prostredím firmy

Zoznámenie sa sprostredím firmy okrem iného slúži aj na pochopenie, prečo jednotlivé časti systému sú tak dôležité pre túto firmu a ako sa môže proces výroby vďaka systému zjednodušiť a urýchliť.

Celý proces sa začína príchodom zákazníka do kancelárie, kde definuje svoje požiadavky na spracovanie svojej zákazky. Zákazník môže priniesť na lakovanie hneď niekoľko zákaziek. Ako jednu zákazku firma vždy uvažuje lakovanie skupiny predmetov práve jednou farbou, alebo jeden predmet rovnako práve jednou farbou. Pre lepšiu predstavu môžme uvažovať skupinu predmetov ako niekoľko desiatok železných respektíve hliníkových profilov. Ako jeden predmet by sme mohli uvažovať bránu rodinného domu. Prvá informácia pre zamestnancov od zákazníka je, aké predmety jeho objednávka zahrňuje. Následne je dôležité špecifikovať farbu. Zákazník môže mať aj špeciálnu požiadavku ohľadom lakovania. Pred tým, ako sa začnú lakovať už prijaté objednávky, musia prejsť niekoľkými úkonmi, aby lakovanie malo zmysel a čo najväčšiu požadovanú kvalitu (viď 2.3). Vzhľadom na to, že firma používa len jednu striekacú kabínu, je v niektorých prípadoch efektívne radiť tie zákazky za seba, ktoré budú lakované rovnakou farbu. V opačnom prípade je potrebné vyčistiť celý lakovací

systém, čo pochopiteľne prináša zdržanie. Po prijatí niekoľkých objednávok, je efektívne vytvoriť plán na niekoľko dní tak, aby zdržanie bolo, čo najmenšie. Zvyčajne si zákazník ešte zadá požiadavku o balení zákazky, ktorá je realizovaná na konci celého procesu. Po procese lakovania sa zákazka uloží do pece a spracuje sa.

Na základe týchto informácii by sme mohli rozdeliť proces výroby do jednotlivých fáz:

- Prijatie a špecifikácia požiadaviek zákazky
- Zaradenie zákazky do plánu výroby
- Príprava zákazky pred lakovaním
- Lakovanie zákazky
- Balenie zákazky

### Pozície zamestnancov vo firme

Predtým, ako budú podrobne popísané jednotlivé fázy výroby, je ešte dôležité spomenúť, aké pozície vykonávajú zamestnanci vo firme.

- Zamestnanec kancelárie - zamestnanci kancelárie príjmaju zákazky od zákazníkov, dohodnú sa s nimi na požiadavkách vrátane ceny a zaradia ju do plánu výroby. Zamestnanci kancelárie majú na starosti aj správu skladových zásob. V procese výroby sa teda zamestnanci zúčastňujú prvých dvoch fáz (prijatie a špecifikácia požiadavok zákazky a zaradenie zákazky do plánu výroby).
- Zamestnanec výroby - zamestnanci výroby na základe plánu výroby postupne spracúvajú jednotlivé zákazky. Začnú fázou prípravy zákazky pred lakovaním a pokračujú poslednou fázou balenia už hotových zakázok.
- Lakovač - ako už samotný názov napovedá, tak tento zamestnanec ma na starosti samotné striekanie zakázok. Okrem lakovania musí aj zaznamenávať množstvo farby, ktoré bolo potrebné na tento proces.

### Prijatie a špecifikácia požiadavok zákazky

Ako bolo spomenuté v úvode kapitoly tak celý proces začína špecifikáciou požiadaviek pre zákazku od zákazníka. V tejto fáze sa zamestnanci dozvedia informácie, ktoré sú nevyhnutné pre spracovanie zákazky a podobne aj doplňujúce informácie, ktoré pomáhajú k naplneniu predstav zákazníka. K najdôležitejším informáciám patrí pochopiteľne upresnenie farby, veľkosť a rozmery zákazky. Medzi základne špecifiká farby patrí RAL (otieň farby), typ povrchu a použitie (interiér alebo exteriér), o ktorých musí podľa vlastného výberu zákazník zamestnancov informovať. Môže nastať situácia, že zákazník nechce použiť farby firmy, ale preferuje úplne iného výrobcu farieb, ako používa lakovacia firma. V takomto prípade musí nastať dohoda medzi zákazníkom a zamestnancami, keďže pre firmu nie je vždy vhodné objednávať farbu pre jednu zákazku. Ak by zákazník trval na svojom výbere, tak musí sám sprostredkovať farbu a opäť informovať zamestnancov o jej antributoch. V takomto prípade firma nemusí evidovať informácie o spotrebe farby. Pre zamestnancov firmy sú rovnako dôležité informácie o lakovaní. Zákazník môže napríklad vyžadovať aj lakovanie so základom. K celkovej kvalite zákazky môžu prispieť doplňujúce informácie o častiach zákazky, ktoré

nie sú vizuálne, respektíve, ktorým netreba venovať veľkú pozornosť a naopak, ktorým áno. Doplňujúce informácie sa môžu týkať tiež použitia už nalakovanej zákazky, ktoré môžu pomôcť k celkovej predstave a priblížiť sa tak k spokojnosti zákazníka.

Všetky tieto informácie je potrebné dodať zamestnancom do výroby od zamestnancov v kancelárii.

### Zaradenie zákazky do plánu výroby

Pri striekaní zákaziek rovnakou farbou by malo byť efektívne ich radiť za seba, keďže v opačnom prípade je potrebné čistenie celého systému. Takéto riešenia zbytočného zdržania nie je vždy pre firmu úplne efektívne. Záleží od ďalších faktorov, ako je dátum potrebného ukončenia zákazky a veľkosť dannej zákazky. Preto firma skôr uvažuje variant spájania zakázok iba v prípade ak je na výrobnej linke ešte priestor pre danú zákazku alebo ak prišli v blízkom čase po sebe do výroby. Firma musí dodržiavať termíny aj ekonomickosť. To sa môže prejaviť, ak je zákazka príliš malá, tak firma sa snaží pretiahnuť termín ukončenia, aby mali možnosť ju spojiť s inou zákazkou rovnakej farby. Pred fázou lakovania je teda dôležité analyzovať zákazky a vyhodnotiť efektívny plán výroby.

### Príprava zákazky pred lakovaním

V tejto fáze zamestnanci firmy pripravia zákazku na lakovanie potrebnými činnosťami (viď 2.3). Každá zákazka potrebuje väčšinou osobitný prístup. V tejto fáze je vhodné poznať farbu zákazky, prípadne dostatok farby na sklade. V prípade nedostatku farby sa túto fázu neoplatí začínať a je vhodný čas a energiu využiť do prípravy inej zákazky, ktorá vyžaduje inú farbu. Do fázy prípravy už zákazník nezasahuje, je to plne na zodpovednosti firmy. Avšak podobne ako informácie od zákazníka, je potrebné doručiť informácie o jej príprave na lakovanie do výroby od zamestnancov kancelárie, ktorí už štruktúru zákazky analyzovali a zaradili do plánu výroby.

### Lakovanie zákazky

Táto fáza závisí od komplexnosti zákazky. Ak by bolo uvažované o striekaní 10 metrov dlhého profilu, tak takéto to striekanie prebehne rýchlejšie a jednoduchšie ako v prípade menších 3D dielcov, kde je potrebné zájsť do každej dutiny. Pri striekaní je potrebné poznať miesta zákazky, ktoré budú viditeľne pri jej využití. Táto fáza sa nemusí vždy podariť na prvý raz. V dôsledku nekvalitného materiálu predmetu, ktorý je lakovaný či v dôsledku nekvalitnej práce v predchádzajúcej fáze alebo pôsobením úplne iného faktora, nemusí farba dobre držať na danom materiály. Pri vyskytnutí problému v tejto fáze sa proces lakovania musí zopakovať. Nie vždy sa jedná o celú zákazku v podobe všetkých predmetov, ktoré zákazka zahrňuje. Môže sa to týkať len tých častí, ktorých sa daný problém dotkol. Ak nastane chyba vo výrobe, tak firma potrebuje zopakovať fázu prípravy, lakovenie a po jej úspešnom ukončení pokračovať poslednou fázou balenia. Pri fáze lakovania zákazky je dôležité spomenúť, že firma sa snaží uchovávať informáciu o spotrebe farby. Ak zamestnanec ukončí fázu lakovania je povinný oznámiť túto informáciu kancelárii, ktorá zabezpečí odpočet aktuálneho množstva farby a je schopná zhodnotiť, ako proces lakovania prebiehal vzhľadom na spotrebu farby.

### Balenie zákazky

Balenie je vnímané zamestnancami ako dôležitá fáza, lebo pri pochybení v tejto fáze sa môže ich produkt počas prepravy poškodiť. Preto je nevyhnutné tieto informácie pripojiť k zákazke a doručiť ich do výroby. Firma zvykne používať balenie, ktoré môžeme nazvať ako štandardné. Môžu sa vyskytnúť aj zákazníci, ktorí vedia, že s ich hotovou objednávkou sa ešte môže niekoľkokrát manipulovať na inom mieste, ako je lakovacia firma. V takom prípade firma k baleniu používa špeciálne prvky, ktoré nie sú súčasťou štandartného balenie. Výroba o takom to štýle balenia musí vedieť.

### Analýza skladových zásob

Analýza skladových zásob nepatrí do procesu výroby, ale je veľmi dôležitá pre chod firmy. Medzi najdôležitejšie štatistiky patrí určite, aktuálne množstvo na sklade, spotreba farby a ktorí zákazníci danú farbu využívajú. Pochopiteľne, ak ma firma lepšiu predstavu o tom koľko sa danej farby využilo za rok, vie objednať väčšie množstvo farby za výhodnejšiu cenu po dohode s výrobcom farieb.

Predstavme si cenové jednanie zákazníka a zamestnanca firmy v kancelárii, kedy zákazník požaduje konkrétnu farbu. Pre pochopenie tohoto príkladu je potrebné si uvedomiť, že pri zaobstaraní veľkého množstva farby pre firmu je cenovo výhodnejšie ako malého množstva. Závisí to ale od dohody firmy a výrobcu farby. Zamestnanec na základe štatistík, ktoré sa týkajú danej farby zistí, že zákazník od nich odoberá len niekoľko desiatok kilogramov danej farby raz za rok a to nepravidelne, ale iný zákazníci z tejto farby využívajú stovky kilogramov ročne. Aby zamestnanec ušetril farbu na sklade pre zákazníkov, ktorí ju s veľkou pravdepodobnosťou využijú, tak zamestnanec ponúkne farbu s rovnakým odtieňom ale možno s inou štruktúrou alebo od iného výrobcu a to so zľavou. Toto by pre firmu mohlo byť výhodne jednanie, lebo nevyužívaná farba sa na sklade eliminuje a veľké množstvo inej farby, ktoré bolo zakúpene výhodne práve kvôli veľkému množstvu sa ušetrí pre stálych zákazníkov.

Firma vďaka štatistikám, ktoré hovoria o tom akí zákazníci využívajú akú farbu sa vie tiež lepšie orientovať vo svojom obore. Napríklad zákazníci, ktorí dodávajú tovar pre firmy v priemysle používajú jednu farbu a zákazníci, ktorí chcú lakovať predmety do exteriéru používajú inú. Potom je pochopiteľne, že firma vie ponúknuť aj poradenské služby pre svojich zákazníkov.

Ďalší príklad prečo je zaznamenavanie štatistík farieb dôležité, sa môže týkať celkovej spotreby farby na danom materiály. Spätnou analýzou spotreby na danú zákazku sa môže zistiť, že farba od iného výrobcu vyžaduje menšie množstvo. Alebo sa zistí možným experimentovaním vo fáze prípravy na lakovanie, že niektoré úkony pomáhajú k tomu, aby farba lepšie držala na danom materiály. Môže nastať aj opačná situácia a v takomto prípade sa firma môže poučiť z vlastných chýb. Štatistika, aké veľké množstvo sa využilo na opravy je často tiež pre lakovacie firmy veľavravná.

## 3.2 Analýza súčasného riešenia

Pre určenie požiadaviek na informačný systém je veľmi potrebné zistiť ako sa jednotlivé činnosti, ktoré bude novovýtvorený systém realizovať vykonávajú v súčasnej dobe.

Informácie o zákazkách momentálne firma uchováva v papierovej podobe. Využíva na to tzv. zákazkový list, kde sa nachádzajú informácie o zákazníkovi a o jeho zákazke, ktorú si

u firmy objednal. Stráca sa možnosť rýchleho vyhľadávania informácii a možnosť rýchleho doplnenia informácií pri zadavaní novej zákazky.

Pri prenose dôležitých požiadavok od zákazníka sa využíva informačná tabuľa, ku ktorej musí zamestnanec kancelárie fyzický prísť. Ak by zamestnanec kancelárie pracoval z domu alebo by sa pracovne nachádzal na inom mieste, musí sa informácia od zákazníka preniesť do výroby telefonátom, cez SMS, alebo iným komunikačným kanálom.

Výsledok analýzy zákaziek a definované plánovanie sa nachádza tiež fyzický na informačnej tabuli. Planovnie človekom je momentálne firmou vnímané ako výhoda vzhľadom na komplexnosť zákaziek, ale nie je vylúčené, že človek nejakú informáciu prehľadne a systém by mohol takúto nepovšimnutú informáciu pripomenúť.

Skladové zásoby sú uchovavané tabuľkovou formou v programe excel. Zamestnanec sa síce dozvie pomerne rýchlym spôsobom aké množstvo farby sa nachádza na sklade a aká je celková spotreba, ale stráca sa automatizovanosť pri zadávaní spotreby a odpočet aktuálne množstvo. Podobne sa stráca aj informácia, kto danú farbu využíva, aké množstvo a ako často.

Zamestnanec, ktorý strieka danú zákazku zaznamenáva množstvo farby v krabici od farby pred striekaním a množstvo po striekaní. Zamestnanci kancelárie potom musia fyzicky počítať rozdiel týchto čísel a zaznamenať odpočet od celkového množstva.

Zamestnanci si často nosia so sebou základné informácie o zákazke alebo o skupine zákaziek (napr. zákazky, ktoré je potrebné spracovať v konkrétny deň) v papierovej podobe. Nie je pre nich vždy pohodlné vyberať mobilné zariadenie a pozerať sa do poznámok. Naviac často sa nechávajú informácie o zákazke v papierovej podobe blízko fyzickej zákazky, aby ten kto s ňou začne manipulovať, vedel o akú zákazku sa jedná a aké úkony je s ňou potrebné vykonať.

## 3.3 Požiadavky na nové riešenie

Jedna z kľúčových požiadavok firmy je prispôsobiť informačný systém na momentálne fungovanie firmy a vychádzať zo súčasného riešenia a jednotlivých fáz výroby (viď 3.1). Zákazník požaduje aj možnosť exportu dát, kvôli potrebe pracovať s nimi aj mimo systému. Ďalšia dôležitá požiadavka je zahrnúť do nového riešenia správu skladových zásob, plánovanie zakázok tak, aby dochádzalo k minimálnemu zdržaniu a ponúkať štatistiky spotreby farieb za účelom úspory. Nové riešenie, ktoré vychádza z požiadavok konkrétnej firmy bolo rozdelené do niekoľkých častí popísaných v tejto podkapitole.

Časti informačného systému:

- Nová zákazka
- Plán
- Sklad
- Zadávanie spotreby
- Kartotéka
- Správa uživateľov

### Nová zákazka

Požiadavka na zadavanie nových zakázok pri ich príjmaní bola taká, aby zamestnanec mohol pohodlne a rýchlo vyplniť údaje o zákazníkovi, ktorý prišiel do kancelárie. Na vyplnenie informácii o zákazníkovi bude stačiť vyplniť len jeho meno a o zvyšné informácie sa už postará systém za predpokladu, že sa nejedná o prvú návštevu zákazníka vo firme.

Údaje o farba podobne predvypĺni systém, ale až po vyplnení dôležitých atribútov farby. Dôležité atribúty farby sú RAL, typ povrchu a použitia. Výrobca farby a kód farby budú vyplnené automaticky, ak sa nachádza na sklade len jeden výrobca. V prípade ak sa na sklade nachádzajú viacerí výrobcovia danej farby, systém sa opýta o akú konkrétnu farbu ma zákazník záujem a ponúkne aj informáciu o aktuálnom množstve. Pre požiadavky od zákazníka a dôležité informácie pre výrobu bol vyhradený samostatný priestor. Tieto informácie sa objavia vo výrobe hneď po zaradení zákazky do plánu. Zamestnanci vo výrobe tak budú mať prístup k informáciám prakticky okamžite.

Jedna z požiadavok je, aby systém upozorňoval na nedostatok farby na sklade. Preto je potrebné, aby ešte predtým, ako bude zákazka zaradená do plánu, zadať odhadované množstvo, aby systém mal predstavu o tom, koľko farby sa využije. Odhadované množstvo je často možné odhadnúť až po premeraní zákazky. Preto je možné zákazku do systému zatiaľ len uložiť a neskôr sa k nej vrátiť. Podobne sa často firme stáva, že zákazníkovi nezáleží na výrobcovi farby a tak si firma zoberie, viac času na premyslenie. V takomto prípade je tiež vhodné zákazku uložiť a priradiť farbu neskôr.

Vloženie novej zákazky počíta aj s tým prípadom, že zákazník, ktorý si objednáva služby firmy sa v systéme ešte nenachádza a podobne, aj farba, ktorú vyžaduje sa v systéme nemusí nachádzať. V dôsledku eliminovania zdržania je, preto navrhnuté riešenie, že v prípade absencie týchto dát v informačnom systéme sa pri zadaní zákazky vytvorí, aj nový zákazník a aj nová farba s nulovým aktuálnym množstvom a požiadavkou na jej objednanie. Môže nastať problém, že farbe budú chýbať atribúty, ktoré zamestnanec pri zadavaní zákazky nemá možnosť zadať. Z časového hľadiska však je výhodný postup, že zamestnanec vybaví zákazníka v kancelárii, zaradí zákazku do plánu, vo firme sa bude vedieť potreba prípravy zákazky a objednania farby. Neskôr kancelária doplní potrebné atribúty farby, ktoré sú potrebné napríklad už pri zadavní spotreby.

### Plán

V novom riešení pre plánovanie sa nachádzajú jednotlivé zákazky s daným poradím. Poradie je určené buď zamestnancom firmy, ktorý ma možnosť meniť poradie zákaziek alebo systémom. Systém ma možnosť navrhnúť zaradenie zákazky za zákazku s rovnakou farbou, aby sa ušetrilo zbytočne zdržanie kvôli čisteniu systému. Navrhnuté zaradenie spočíva v tom, že novovytvorenej zákazke bude pridelené nové miesto za prvú skupinu zákaziek od začiatku plánu s rovnakou farbou. Prvú nájdenú skupinu, preto, lebo ak zamestnancom sa nepodarí nájsť miesto na výrobnej linke pre zákazku (komponenty zákaziek sa vešajú na akýsi vozík), ktorá bola radená, tak ešte stále je možnosť využiť inú skupinu zákaziek. Ak by sa zákazka zaradila k zákazke, čo najďalej v pláne, tak firma by mohla prísť o príležitosť spojenia zákaziek s prvou skupinou. Zamestanec musí tento návrh systému pri zadavaní zákazky vždy potvrdiť vzhľadom na to, aby zamestnanec nad plánovaním nestratil kontrolu. Systém teda upozorní zamestnanca na možnosť eliminovať zdržanie pri výrobe, ale rozhodnutie ostáva na samotnom zamestnancovi kancelárie. Zákazky v pláne obsahujú nie len informácie o farbe, ale aj konkrétne požiadavky na zákazku. V zákazkách sa nachádza

stav zákazky a aj upozornenie na nedostatok farby, aby sa eliminovala zbytočná príprava zákazky pred lakovaním.

V tejto časti systému bola navrhutá aj možnosť meniť stav zákaziek. Stav zákazky môžu nadobudnúť hodnotu *hotovo, oprava* alebo *čiastočná oprava*. Zákazka zmizne z plánu len v prípade zmeny stavu na *hotovo*. V prípade zmeny stavu na *oprava*, systém počíta s opravou celého množstva zákazky a v prípade *čiastočnej opravy* je to polovičné množstvo. Systém totiž musí počítať s ďalšími výdavkami, ale zamestnanec výroby, ktorý bude zväčša meniť stav zákaziek, nemá čas prepočítavať aké konkrétne množstvo bude potrebné na opravu.

### Sklad

Nové riešenie skladu sa oproti súčasnému riešeniu v exceli, vzhľadom na ponúknuté informácie veľmi nemení. Je ale doplnený o štatistiky jednotlivých farieb. Tieto štatistiky hovoria o tom, aký zákazník využil aké veľké množstvo farby za určité obdobie vrátane toho aké množstvo sa využilo na opravy. Sklad bude upozorňovať zamestnancov aj o pomerne malom množstve farby na sklade. Hranicu si môžu určiť zamestnanci podľa vlastného uváženia. Podobne sklad vizuálne vyznačí zostatok farby aj v situácii, keď je momentálne rezervované množstvo pre farbu väčšie, ako je aktuálne množstvo farby. V takamto prípade je veľmi pravdepodobné, že aktuálny zostatok na lakovanie nemusí stačiť.

### Zadavanie spotreby

Požiadavka na zadavanie spotreby bola vytvoriť vlastné užívateľské rozhranie pre zamestnanca zastupujúcu rolu lakovača. Nové riešenie spočíva v tom, že dochádza k automatizovanému zadavaniu spotreby a odpočitavania aktuálneho množstva na sklade.

Keďže zistenie danej spotreby sa počíta vážením farby pred lakovaním a po lakovaní, nové riešenie počíta aj so situáciou, že počas lakovania došlo k výmene krabice, pretože pôvodná neobsahovala dostatočné množstvo. Aj s touto situáciou systém počíta. Systém zistí veľkosť krabice a odpočíta namerané množstvá. Napríklad, ak pôvodná krabica obsahovala už len tri kilogramy, došlo k výmene farby za novú krabicu s kapacitou tridsať kilogramov, po výmene a nastrekaní nová krabica obsahovala 28 kilogramov, tak systém v konečnom dôsledku vypočíta spotrebu 5 kilogramov. Zamestnanec sa nemusí zaťažovať výpočtami a stačí mu len vážiť množstvá. Systém potrebuje vedieť aj váhu krabíc od danej farby, aby nameraná spotreba bola čo najpresnejšia. Váha krabice sa počíta len v gramoch, ale v konečnom dôsledku pri lakovaní väčšieho množstva to môže znamenať veľký rozdiel, ktorý môže byť smerodajný v následnej analýze skladových zásob.

Zadavanie spotreby bude uľahčovať prácu zamestnancom aj pri lakovaní viacerých zákaziek. Namerané spotrebované množstvo sa bude vážiť rovnakým už spomenutým spôsobom, avšak môže nastať problém pri pridelení skutočného množstva spotreby pre zákazku. Je zjavné, že pri lakovaní len jednej zákazky, systém vie určiť spotrebu úplne presne. Pri lakovaní dvoch a viac zákaziek už nastáva drobná odchýlka v zadanej spotrebe. Je pochopiteľné, že podnik viac zaujíma ušetrený čas pri lakovaní a zadavní spotreby ako presne namerané množstvá. Tento problém bude riešiť systém tak, že sa vypočíta pomer odhadovaného množstva zákaziek a aplikuje sa na odváženú celkovú spotrebu daného lakovania. Takto sa elimuje odchýlka pri meraní a ušetrí sa celkový čas.

### Kartotéka

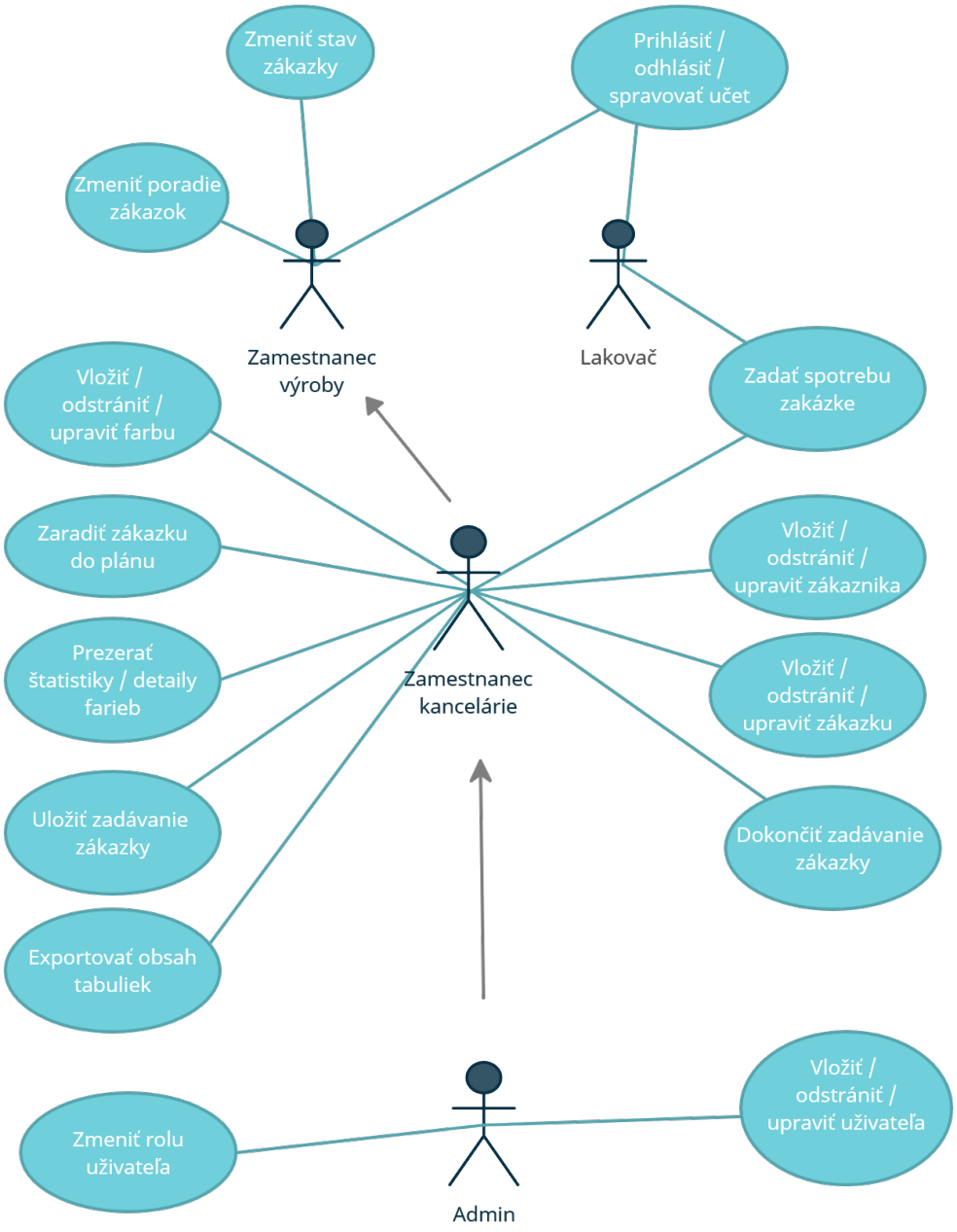
Časť informačného systému s názvom kartotéka vychádza z požiadavky uchovávať históriu zákaziek a údajov o zákazníkovi. Riešenie je rozdelené do dvoch častí, a to práve pre vyhľadávanie zakázok a zákazníkov.

### Správa uživateľov

Táto časť nevychádza zo žiadnej požiadavky, ale je potrebné rozdeliť užívateľov systému podľa ich pozícii vo firme, a to pochopiteľné vyžadaju aj správu užívateľov. Zamestnanec kancelárie bude mať k dispozícii všetky rozhrania okrem správy užívateľov. Tú bude mať k dispozícii iba majiteľ firmy. Pre prácu zamestnanca výroby je postačujúci len plán a práca lakovača bude zahŕňať len zadávanie spotreby.

## 3.4 Diagram prípadov použitia

Jedným z výstupov špecifikácie požiadavkou je väčšinou **diagram prípadov použitia** (angl. Use Case diagram). Kľúčovými aktivitami vo fázy špecifikácie požiadavkov je hľadanie **prípadov použitia** a ich **účastníkov**. Jeden prípad použitia je chápaný ako funkcia, ktorú systém vykonáva menom jednotlivých účastníkov alebo v ich prospech. [18] Pri pozeraní na diagram je okrem funkcií vhodné upozorniť aj na roly jednotlivých užívateľov, ktoré sa nelíšia od reality (viď 3.1) s výnimkou roly admina, ktorý v systéme nesmie prirodzene chýbať.




Obr. 3.1: Diagram prípadov použitia vyvíjaného informačného systému.

# Kapitola 4

# Návrh informačného systému

Predchádzajúca kapitola bola venovaná požiadavkam firmy. Teraz, ak sú požiadavky známe a je možné si predstaviť, čo je úlohou systému, tak ďalším krokom môže byť pristúpenie k návrhu databázy, kde budú predstavené konkrétne dáta, ktoré so systémom pracujú. Ako dátový návrh je v tejto kapitole použitý ER diagram, ktorý je neskôr potrebné previesť do tabuliek relačnej databázy. Bude predstavené aj užívateľské rozhranie, čo môže pomocť dosiahnuť konečnú predstavu o systéme.

## 4.1 ER diagram

Pri predstavení dát v systéme si pomôžeme diagramom entít a vzťahov (ang. Entity Relationship Diagram), ktorý slúži k modelovaniu dát aplikačnej domény a ich vzťahov "v kľude". ER diagram je entitno vzťahový model popisujúcí návrh uložených dát v systému na vyššej úrovni abstrakcie. ER diagram modeluje dáta, ktoré potrebujeme v systéme uchovávať a vzťahy medzi týmito dátami. Používa sa ako základný model dátového modelovania.

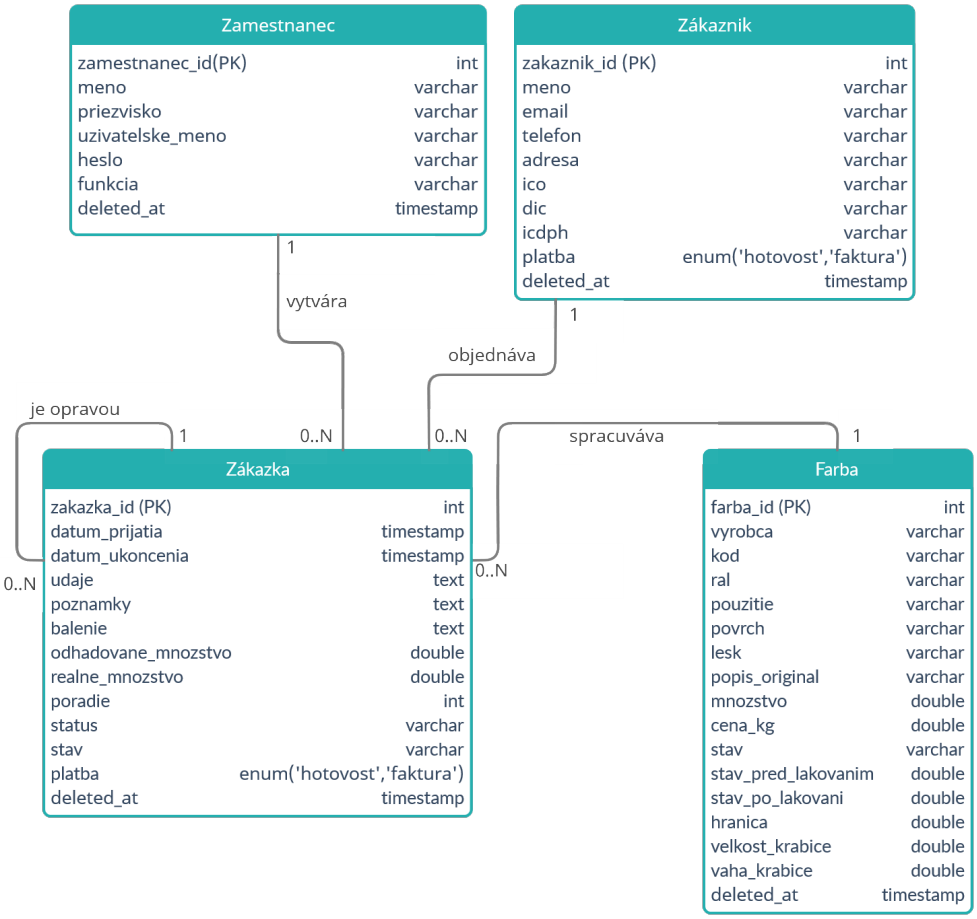
Takýto model využíva tzv. **Entity**. Entita je "vec"reálneho sveta (vec býva často označovaná ako objekt). Každá taká entita musí byť rozlíšiteľná od iných entít (musí byť jedinečná). Príkladom entity môže byť konkrétny zamestnanec firmy (každého zamestnanca vieme od ostatných odlíšiť). **Entitná množina** je množina entít rovnakého typu, ktoré zdieľajú rovnaké vlastnosti (atribúty).

**Vzťah** je vedľa entity (entitnej množiny) ďalším dôležitým prvkom v ER diagrame. Vzťah vyjadruje asociáciu medzi entitami. To je situácia, kedy dve prípadne viac entít spolu logicky súvisia. Napríklad, ak k zákazke je priradená farba. Medzi týmito dvoma entitami je vzťah. **Vzťahová množina** je množina vzťahov rovnakého typu, ktoré zdieľajú rovnaké vlastnosti.

Ďalším základným prvkom sú **atribúty**. Atribút je vlastnosť entity, ktorá nás v kontextu daného problému zaujíma. Atribút je definovaný ako prvok entitnej množiny (napríklad u entitnej množiny zákazník nás zaujíma ID klienta, meno, priezvisko, adresa, telefónne číslo a pod.). Atribúty môžu byť jednoduché alebo zložené, jednohodnotové alebo viachodnotové a prázdne. Ak sú atribúty prázdne atribúty môžu nadobúdať špeciálnu hodnotu NULL. [18]

V nasledujúcej časti podkapitoly je popísaný dátový návrh vyvíjaného informačného systému v podobe ER diagramu a popis jednotlivých entitných množín a vzťahov. Každá tabuľka obsahuje okrem uvedených atributov aj atribut *deleted_at*. Je využitý princip tzv."soft deleting", ktorý umožňuje označiť niektoré záznamy ako odstránené bez skutočného vymazania z databázy. Účinne sa tak zabráni výberu záznámu a stále môže byť odkazovaný do

iného záznamu (napr. ak sa odstráni zákazník zo systému, stále je dobre pre firmu vedieť, kto danú zákazku požadoval).




Obr. 4.1: Diagram entít a vzťahov vyvíjaného informačného systému.

### Entitná množina Zákazník

Entitná množina **zákaznik** v podobe svojich atribútov uchováva základné osobné údaje o jednotlivých zákazníkoch, ktorí dávajú alebo dali svoje produkty spracovať lakovacej firme.

Všetky atribúty okrem *identifikátora* a mena zákazníka môžu nadobúdať nulovej hodnoty, a to z dôvodu, aby sa zákazník mohol zaevidovať do systému aj bez údajov, ktoré mu môžu byť doplnené neskôr. Užívateľ systému (zamestnanec firmy) tak nebude zdržiavaný a môže mu vytvoriť zákazku aj napríklad bez telefónneho čísla alebo e-mailovej adresy.

Atribúty *IČO* (identifikačné číslo organizácie), *DIČ* (daňové identifikačné číslo) a *IC DPH* (identifikačné číslo pre daň z pridanej hodnoty) môžu nadobúdať nulovú hodnotu dokonca aj počas celého bytia zákazníka v systéme, keďže sa môže nachádzať v systéme, aj súkromná osoba, ktorá týmito hodnotami nedisponuje. Množina zákazník ešte uchováva atribút *platba* , ktorý slúži len na to, aby system bol schopný predvoliť preferovaný spôsob platby zákazníka.

Zákazník sa môže nachádzať v systéme aj bez zákazky, avšak zákazku bez zákazníka nie je možné vytvoriť.

### Entitná množina Zákazka

Entitná množina **zákazka** uchováva údaje o zákazke. Okrem *dátumu prijatia* a *dátumu ukončenia* množina uchováva aj atribúty *údaje* a *balenie*, čo predstavuje väčšinou rozmery zákazky a heslovité požiadavky od zákaznika, ktoré sa zobrazia vo výrobe zamestnancom výroby (viď 3.1). Atribút *poznámky* sú interné poznámky firmy väčšinou pre zamestnancov kancelárie (napr. zľava pre zákazníka). *Odhadované množstvo* pomáha systému mať predstavu o stave farby pred lakovaním a *reálne množstvo* zase o spotrebe farby. *Poradie* je prirodzené číslo pridelené k zákazke, ktorá sa nachádza v pláne. Atribút *stav* predstavuje textovú hodnotu, ktorá hovorí či je zákazka hotová alebo naopak ešte vo výrobe. K tomu či hotovej zákazke predchádzali opravy napovedá atribút *status*.

K zákazke systém neumožňuje prideliť viac ako jedného zákazníka, ale pochopiteľne zákazníkovi môžu byť pridelené niekoľko zákaziek. Zákazka vždy disponuje práve jednou farbou (viď 3.1). Zákazku vytvorí jeden zamestnanec kancelárie.

Za zmienku ešte stojí unárna vzťah, ktorý vyjadruje, že zákazka môže odkazovať na inú zákazku v prípade ak bola jej opravou.

### Entitná množina Zamestnanec

Túto entitnú množinu **zamestnanec** by sme mohli pomenovať aj užívateľ. Ide totiž o zamestnancov, ktorí budú so systémom pracovať. Typicky pre užívateľov v systéme je potrebné uchovávať *užívateľské meno* a *heslo*. Atribút *funkcia* slúži na to, aby systém vedel, aké užívateľské rozhranie zobraziť. Zamestnanec má vzťah iba k zákazke, ktoré môže vytvoriť niekoľko.

### Entitná množina Farba

Entitná množina **farba** ma niekoľko dôležitých atributov pre chod systému. Trojice atribútov *ral*, *povrch* a *použitie* sú atribúty, ktoré sú pre zamestnancov veľavravné. Hodnota atributu *ral* je väčšinou štvorčíslie zo štandardu vzorkovníka farebných odtieňov farieb, ktoré sa používajú v priemyselnej oblasti. *Povrch* hovorí o štruktúre farby a použitie o tom či je farba určená do exteriéru alebo interiéru. Na to, aby farba mohla byť teoretický jednoznačne indetifikovaná potrebujeme ešte *výrobcu* farby a *kód*, ktorý si určuje výrobca farieb. Systém však na identifikáciu používa iný identifikátor, ktorý uľahčuje prácu pri zmene už spomenutých atribútov (napr. zmena je potrebná pri preklepe zadávaní atribútov farby).

Atribúty ako *lesk*, *popis_original* (označenie farby v angličtine) a *cena_kg* (cena za kilogram) sú pre zamestnancov čisto informatívne a evidujú ich skôr pre úplnosť.

Jeden z ďalších atribútov, ktoré sú väčšinou potrebné pre správny chod systému je *stav*, ktorý je nevyhnutné evidovať, ak je farba uložená len dočasne. *Velkost_krabice* a podobne aj *vaha_krabice* sú potrebné pri počítaní spotreby. Atribút *stav_pred_lakovanim* spolu s ďalším atribútom *stav_po_lakovaním* slúžia tiež pri počítaní spotreby, kedy je potrebné uchovávať množstvo hlavne pred lakovaním, ak by náhodou nastalo počas lakovania vypnutie systému. Atribút *hranica* slúži na upozornenie zamestnancov, pri prekročení tejto hodnoty farby na sklade.

Farba ma vzťah pochopiteľne k zákazke, kde farba môže byť pridelená k niekoľkým zákazkám.

## 4.2 Návrh uživateľského rozhrania

Návrh užívateľského rozhrania je smerodajným a veľmi dôležitým faktorom pri rozhodovaní o tom, aby výsledná webová aplikácia bola pre užívateľov jednoduchá a užívateľský prívetivá na používanie. Návrh grafického užívateľského rozhrania a diskusiu spojenú s ním je vždy dobre zahrnúť ešte pred samotnú implementáciu. Grafický návrh bol diskutovaný s majiteľom lakovacej firmy a prispôsobený jeho predstavám. K tomu, aby sa návrh užívateľského rozhrania prispôsobil zakazníkovým predstavám je často dobre použiť mockup. Mockup efekt sa zvyčajne využíva k prezentovaniu výslednej práce v realistickom podaní. Pomáha vývojárom a ich zákazníkom si lepšie predstaviť grafický výsledok takmer finálneho produktu.

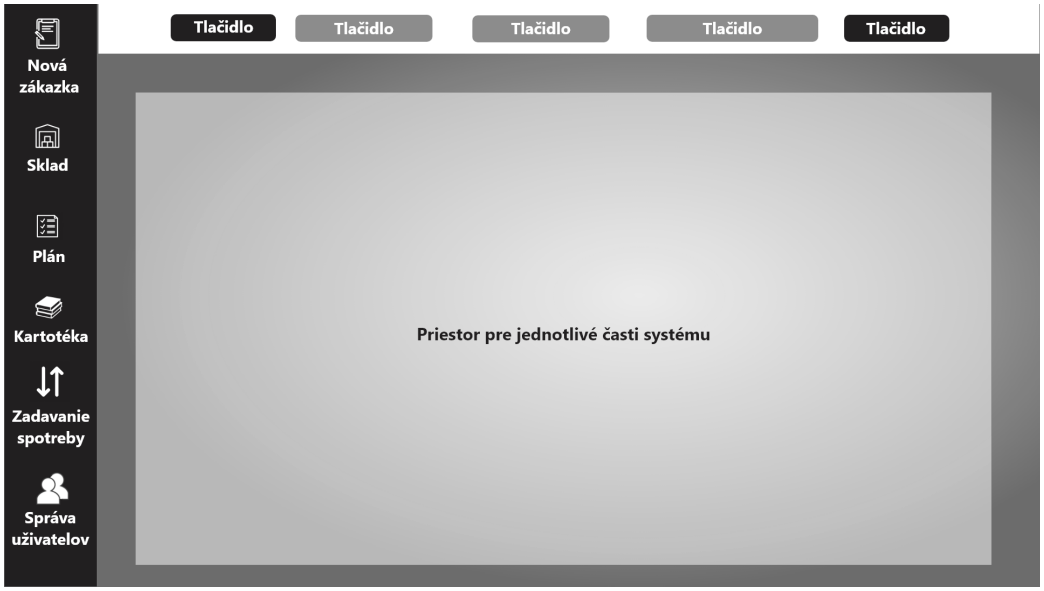
Návrh uvažuje rozdelenie informačného systému do častí, ako to bolo popísané v kapitole 3.3. V systéme bude teda zadavanie novej zákazky s prípadnou úpravou zákaziek, plán výroby, sklad, kartotéka s históriu zákaziek, zadavanie spotreby a správa užívateľov. V tejto podkapitole sa nebude už hovoriť o tom, čo bude systém v jednotlivých častiach robiť, skôr budú len načrtnuté výsledky diskusie o grafickom užívateľskom rozhraní.

Je vhodné ešte spomenúť a ujasniť si, že užívatelia informačného systému nebudú stovky ľudí za hodinu, ale možno len desiatka prístupov denne, a to vždy rovnakí užívatelia, ktorí predstavujú zamestnanci firmy. Je predpokladané, že užívateľ sa so systémom zoznámi po pár dňoch používania. Preto bol zvolený prístup, neklásť príliš veľký dôraz na intuitívnosť rozhrania, ale skôr na funkcionalitu.

Makety boli vytvorené pomocou programu *Adobe XD* (tiež známy ako Adobe Experience Design), ktorý je vektorový nástroj a umožňuje vytvárať návrhy grafických užívateľských rozhraní webových alebo mobilných aplikácií, vyvinutý a publikovaný spoločnosťou Adobe Inc.

### Návrh hlavnej šablóny

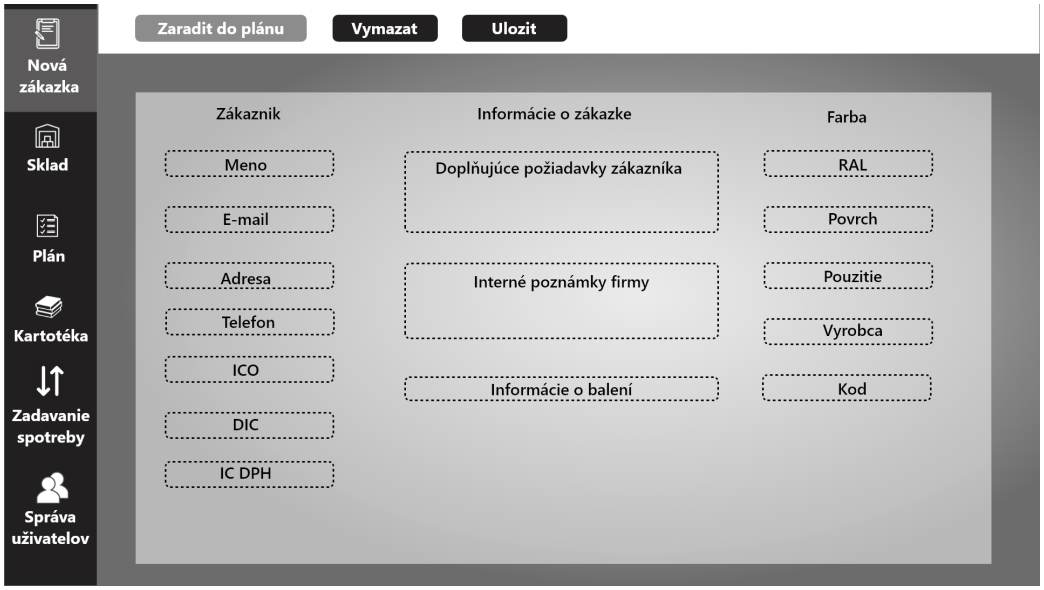
Na obrázku 4.2 je vidieť návrh hlavnej šablóny, ktorá pozostáva z troch častí. Prvá časť je menu na ľavej strane obrazovky, kde si užívateľ môže vybrať z jednotlivých časti systému, ktorú by si prial spravovať. Správa vybranej časti, je doplnená o funkcionalitu tlačidlami, ktoré sa nachádzajú na vrchu obrazovky. Počet a funkcionalita tlačidiel závisí od vybranej časti informačného systému. Podobne od výberu užívateľa v menu závisia aj elementy, ktoré sa zobrazia v priestore, ktorý je na obrázku nazvaný ako *priestor pre jednotlivé časti systému*.




Obr. 4.2: Návrh hlavnej šablóny grafického užívateľského rozhrania.

### Návrh šablóny pre tvorbu zákaziek

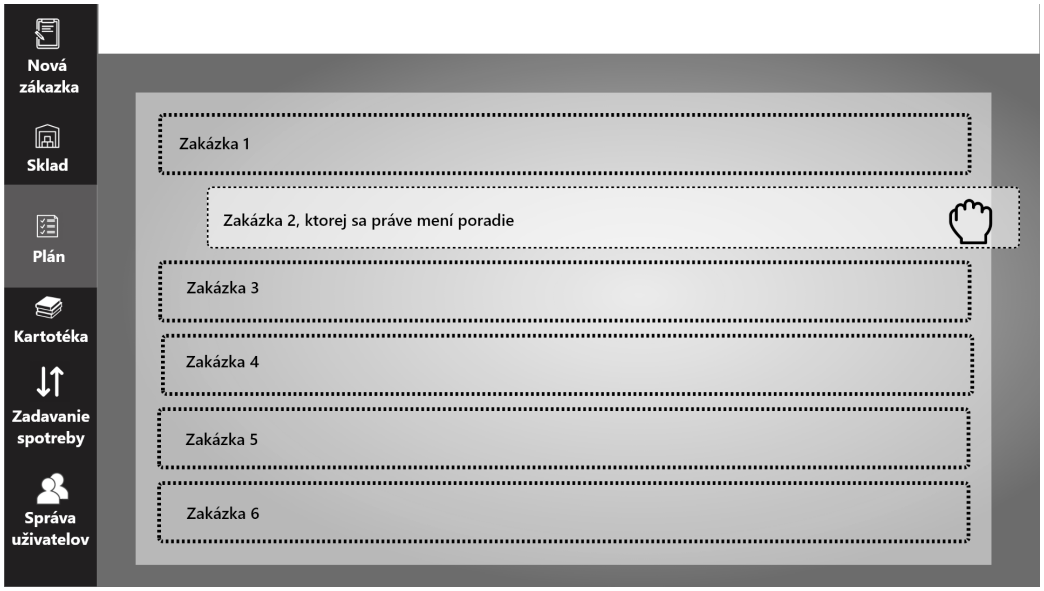
Na obrázku 4.3 je možné si všimnúť návrh šablóny pre tvorbu nových zákaziek. Pre zamestnanca firmy bolo dôležité vidieť všetky atribúty, ktoré potrebuje zadať pri príchode zákazníka do kancelárie. To nie je možné zabezpečiť na mobilných zariadeniach, ale pri návrhu bolo myslené aj na rezponzívny dizajn. V šablóne si je možné všimnúť informácie o zákazníkovi, ktoré vyplní systém alebo samotný užívateľ, informácie dôležité pre výrobu a informácie o farbe.




Obr. 4.3: Návrh šablóny grafického užívateľského rozhrania pre tvorbu nových zákaziek.

### Návrh šablóny pre plán výroby a zadávanie spotreby

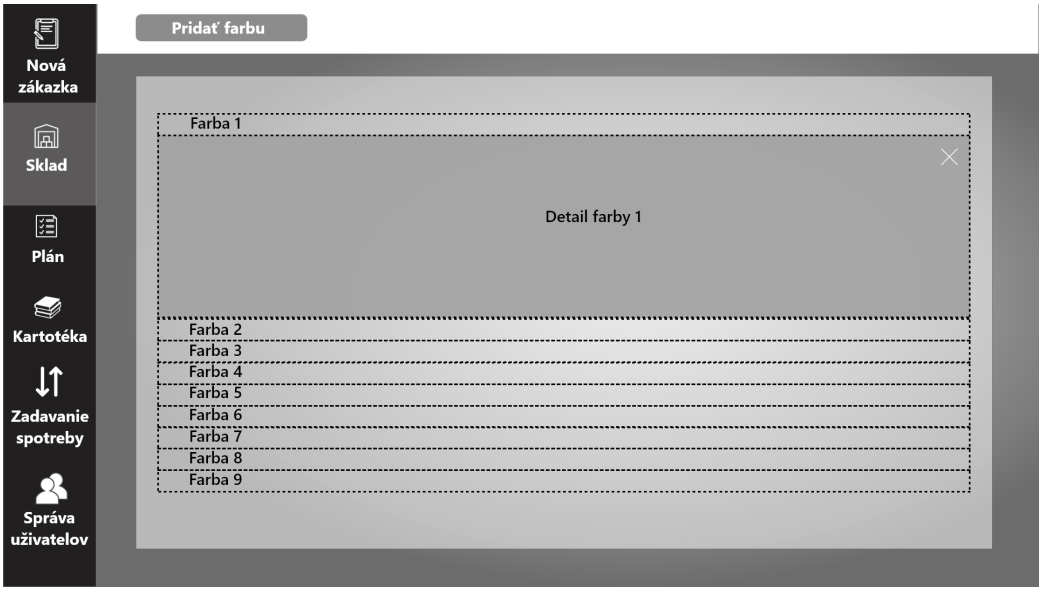
Obrázok 4.4 ukazuje plán výroby, kde je možné si všimnúť riešenie zmeny poradia zákaziek metódou *"drag and drop"*. Návrh užívateľského rozhrania pre zadávanie spotreby vyzerá takmer rovnako, ale bez informácií, ktoré nie sú pre lakovača podstatné a bez možnosti meniť poradie zakázok.




Obr. 4.4: Návrh šablony grafického užívateľského rozhrania pre plán výroby.

### Návrh šablóny pre tabuľkové riešenia

Obrázok 4.5 síce ukazuje návrh šablóny pre sklad, ale pre časti systému, ako je kartotéka alebo správa užívateľov je riešenie tiež v podobe tabuľky. Skladové zásoby sú ale doplnené o možnosť si "rozkliknúť"detail farby a vidieť zaujímavé štatistiky v podobe grafov alebo tiež informácie o zákazkách, s ktorými farba súvisí. V prípade malej obrázovky je tiež možné kliknutím na riadok tabuľky vidieť stĺpce, ktoré sa na obrazovku už nezmestili. Tento rezponzívny dizajn je možný vďaka nastaveniam v Datatables od knižnice jQuery.




Obr. 4.5: Návrh šablóny grafického užívateľského rozhrania pre skladové zásoby.

# Kapitola 5

# Použité technológie

V tejto kapitole budú predstavené technológie použité pre tento projekt. Najkôr jednotlivé programovacie jazyky, ako sú PHP pre serverovú časť alebo HTML, CSS a Javaskript pre klientskú. Na konci kapitoly je z ľahká rozobratý framework Laravel, ktorý uľahčil implementáciu tohto projektu.

## 5.1 PHP

PHP je skriptovací jazyk na strane servera, ktorý bol navrhnutý špeciálne pre web. Do stránky HTML sa môže ľahko vkladať kód jazyka PHP, ktorý sa vykonáva pri každej návšteve tejto stránky. Kód jazyka PHP je interpretovaný webovým serverom, pričom generuje dokument HTML alebo iný výstup, ktorý uvidí návštevník. PHP je projekt s otvoreným zdrojovým kódom, čo znamená, že exituje voľný prístup k jeho zdrojovému kódu. Je možné ho používať, meniť a distribuovať. [22] Aktuálna hlavná verzia jazyka PHP je verzia 7, ktorá bola použitá aj v tomto projekte.

### Prednosti PHP

Medzi hlavných konkurentov jazyka PHP patri jazyk Python, Perl, Java a pod. V porovnaní s týmito jazykmi ma však PHP niekoľko predností.

- Výkon - je veľmi rýchli, s pomocou jedného lacného serveru môže spracovať milióny návštev denne
- Škalovateľnosť - je možné ho efektívne a ľahko implementovať do veľkého množstva webových serverov
- Pomerne ľahké učenie a použiteľnosť - syntax jazyka vychádza z iných programovacích jazykov - najmä jazyka C a Perl, keď už existuje u niekoho znalosť týchto jazykov tak je väčšia pravdepodobnosť, že bude produktívny aj v jazyku PHP
- Vstavané knihovne - jazyk PHP bol navrhnutý pre použitie na webe a obsahuje množstvo vstavaných funkcií, ktoré vykonávajú veľa užitočných webových úloh
- Silná podpora objektovo orientovaného programovania - obsahuje skvelé navrhnuté objektové orientované prvky - dedičnosť, súkromie, chránené vlastnosti a metody a pod.

- Dostupnosť zdrojového kódu a dokumentácie - ak niekto chce niečo pridať do jazyka, môže tak urobiť, netreba čakať na novú verziu, jazyk PHP má takisto vyspelú dokumentáciu a komunitu

  [22]

## 5.2 MySQL

MySQL je relačný databázový server, ktorý ponúka rovnaké funkcie ako konkurenčné produkty. Inými slovami, príliš veľa prekvapení sa v MySQL stretnúť nedá, ak existuje skúsenosť s iným databázovým produktom. [17] Existuje však stále množstvo dôvodov prečo pozerať na databázový systém MySQL ako dobrú voľbu:

- Výkon - databázový systém MySQL je nepopierateľne jeden z najrýchlejších
- Ľahkosť použitia - ak už u užívateľa existuje skúsenosť so systémom pre správu relačnej databázy, nemal by mať žiadne ťažkosti ani s MySQL, ktorý je ešte jednoduchšie nastaviť než väčšinu jemu podobných produktov
- Prenositeľnosť - MySQL je možné používať v unixových operačných systémoch a tiež v systéme Windows
- Dostupnosť zdrojového kódu a dokumentácie - podobne, ako u jazyku PHP je možné prispieť do zdrojového kódu, avšak táto výhoda pre množstvo programátorov nie je dôležitý faktor, ale určite výhodou je istota, že vývoj bude fungovať s veľkou pravdepodobnosťou aj v budúcnosti

  dostupná je aj pomerne pestrá dokumentácia a množstvo návodov

  [22]

## 5.3 HTML

Zjednodušene povedané, webová stránka (alebo dokument HTML) je obyčajný textový súbor, ktorý bol kódovaný pomocou jazyka Hypertext Markup Language (HTML), aby vyzeral pekne naformátovaný vo webovom prehliadači.

Kód v súbore HTML pozostáva z textu obklopeného značkami. Tieto značky označujú, kde by sa malo formátovanie použiť, ako by malo vyzerať rozloženie, aké obrázky by mali byť umiestnené na určitých miestach a pod. [23]

## 5.4 CSS

Webových dizajnérov, ktorí pri vytváraní rozsiahlych webových stránok pracovali so skoršími verziami HTML, často frustrovalo množstvo opakovaní, ktoré súvisia s ich prácami. Predpokladajme, že celá webová stránka má 200 HTML dokumentov, všetky s rovnakým základným usporiadaním a dizajnom. Aby bolo možné vykonať zmenu dizajnu na celom webe, musel by návrhár vstúpiť a každú z týchto 200 dokumentov ručne upraviť. Neskôr sa to podarilo obísť vďaka neskorším verziám HTML, ktoré podporovali kaskádové štýly. [23] Za pomoci pravidiel kaskádových štýlov, sa dizajnovanie stránok prenieslo na inú úroveň. Používajú tzv. selektor, ktorý vyberie HTML element alebo skupinu elementov a pomocou pravidiel sa im nadefinuje určitý vzhľad.

## 5.5 Bootstrap

Bootstrap je najpopulárnejší HTML, CSS a JavaScript framework pre vývoj responzívných webových stránok. [2] Responzívny dizajn umožňuje webovej stránke alebo aplikácii zistiť veľkosť a orientáciu obrazovky návštevníka a podľa toho automaticky prispôsobiť zobrazenie. Bootstrap ponúka jednotlivé komponenty alebo rovno celú šablónu týchto komponentov, ktoré sa dajú jednoducho použiť pri programovaní.

## 5.6 Javascript

Je jeden z najpopulárnejších skriptovacích jazykov použiteľných v rámci HTML dokumentov. Je dynamický typovaný a funkcie sú v ňom k dispozícii ako bežný dátový typ. Jeho syntax vychádza z jazyka C a niektoré standartné rozhrania (vstavané funkcie, objekty, premenné) sa podobajú jazyku Java. [24] Taktiež sa pomocou javascriptu môžu programovať rôzne animácie a v kombinácii CSS a HTML predstavujú technológie, bez ktorých sa dnešný moderný web predstavuje len veľmi ťažko. Často sa na zjednodušenie Javascriptu používa jeho knižnica jQuery.

### JQuery

JQuery je rýchla, jednoduchá a na funkcie bohatá knižnica JavaScriptu. Sila jQuery je prístupná cez JavaScript, takže jeho pochopenie je pri používaní jQuery nevyhnutné. Vďaka ľahko použiteľnému rozhraniu API, ktoré funguje vo veľkom množstve prehľadávačov, je manipulácia s dokumentami HTML, manipulácia s udalosťami alebo volanie AJAX (Asynchronous JavaScript And XML) oveľa jednoduchšie. [11] Účelom jQuery je uľahčiť používanie JavaScriptu pri programovaní webových stránok alebo aplikácií. JQuery pri bežných úlohách, ktoré vyžadujú veľa riadkov kódu JavaScriptu, zoberie kód a zaobalí ho do metód, ktoré je možné zavolať pomocou jediného riadka kódu. [12]

### Datatables

DataTables je doplnok pre knižnicu Javascript jQuery. Jedná sa o vysoko flexibilný nástroj, ktorý je založený na základoch postupného vylepšovania. Snaží sa teda pridávať všetky pokročilé funkcie do akejkoľvek tabuľky HTML. Koncoví používatelia musia byť schopní, čo najrýchlejšie získať užitočné informácie z tabuľky, a preto majú Datatables zabudované funkcie, ako je filtrovanie, vyhľadávanie a stránkovanie.

Široká dokumentácia Datatables užívateľom poskytuje informácie, ktoré potrebujú na to, aby mohli vo svojich aplikáciách prispôsobovať svoje tabuľky zodpovedajúce ich presným požiadavkám. Na spojazdnenie Datatables stačí len zahrnúť do projektu pár skriptov a zavolať jednu funkciu na vytvorenú tabuľku. Potom si užívateľ môže prispôsobiť tabuľku podľa svojich predstav použitím rôznych nastavení. [8]

### Chart.js

Chart.js je voľne dostupná Javascript knižnica, ktorá ponúka jednoduché, ale flexibilné mapovanie JavaScriptu pre návrhárov a vývojárov. Dokáže vizualizovať údaje v podobe až ôsmich grafoch a každý z nich je animovaný a prispôsobiteľný. Zabezpečuje vynikajúci výkon vykreslenia vo všetkých moderných prehľadávačoch a dokáže prekresliť graf na veľkosť okna, aby bola dosiahnutá dokonalá mierka. [4]

## 5.7 Laravel

Frameworky sa dnes už pri zložitejších projektoch používajú bežne, a to z jednoducho dôvodu. Tento dôvod by sa dal prirovnať napríklad k balíkom alebo komponentám, ktoré sú k dispozícii vývojárom jazyku PHP, kde je niekto iný zodpovedný za vývoj a údržbu izolovaného kódu, ktorý ma presne definovanú funkciu. Teoreticky má táto osoba hlbšie pochopenie tejto súčasti ako užívateľ, ktorý ju využíva. Frameworky, nie len ako Laravel, ale aj, ako Symfony či Nette obsahujú zbierku komponentov, ako sú konfiguračné súbory, adresárové štruktúry alebo iné poskytované služby spolu s prispôsobeným frameworkom.

Výhodou používania frameworku je vo všeobecnosti, že niekto urobil nie len rozhodnutia o jednotlivých komponentách, ale aj o tom, ako by tieto komponenty mali do seba zapadať. Laravel tiež pomáha vývojárom webových stránok zjednodušiť ich vývojový proces pomocou čistého a opakovane použiteľného kódu. Je to jeden z frameworkov, ktorý má flexibilitu a bohaté funkcie, ktoré ho robia dokonalou platformou pre vytváranie webových stránok a aplikácií. Laravel prichádza s architektonickým vzorom model-view-controller (MVC). Ľahko sa používa a ponúka tak mimoriadne pohodlný spôsob vytvárania veľkých alebo malých podnikových aplikácií. Vďaka tomu je ľahšie organizovať väčšie a zložitejšie projekty. Programovacie úlohy často zahŕňajú použitie šablónového nástroja, ktorý funguje ako nástroj na spracovanie veľkého množstva textových dát vo webových aplikáciách. Stručne povedané, šablónový modul spája dátový model, spracováva kód uvedený v zdrojových šablónach a smeruje výstup do konkrétneho textového súboru. Je možné použiť hlavnú šablónu na vytvorenie jednoduchého rozloženia, ktoré je možné rozšíriť o ďalšie súbory. Skvelou výhodou, ktorá poteší hlavne vývojarských začiatočníkov, je rozsiahla dokumentácia a množstvo návodov. [16] Z týchto jednoduchých dôvodov a aj z dôvodu skúsenosti autora s frameworkom Laravel, bol práve tento PHP framework vybratý pre tento projekt. Ďalšie alternatívy okrem frameworku Laravel, u ktorého sa takmer všetko konfiguruje v PHP, by mohli byť populárne hlavne v Českej republike už spomínané frameworky Symfony, kde sa veľa vecí dá riešiť deklarativne a netreba niektoré časti konfigurovať procedurálne v PHP. Ďalšia vhodná alternatíva by mohol byť framework Nette, ktorý je trošku menši než Laravel alebo Symfony, ale postupne sa vyvíja, zvládne toho už naozaj veľa a je sprevádzany českou dokumentáciou a komunitou českých vývojarov, ktorí su vždy ochotní poradiť.

V ďalšej časti tejto krátkej podkapitoly o použitých technológiách bude spomenutých pár ďalších súčasti Laravelu, ktoré urobili tento projekt jednoduchším pri jeho implementácii. Väčšina informácii je čerpaných z knihy Laravel: Up & Running [20] ak nie je uvedené inak.

### Composer

Akýkoľvek stroj, na ktorom je vývívajaný projekt za pomoci Laravelu by mal byť nainštalovaný Composer. Je to nástroj, ktorý je základom najmodernejšieho vývoja PHP. Composer je správca závislostí (napr. knižníc, ktoré riešia určité problémy) a je základom mnohých testov, načitaných skriptov, inštalačných skriptov a pod. Composer je skrátka potrebný pre všetky externé závislosti a je potrebný aj pri inštalácii Laravelu a pri jeho aktulizáciach.

### MVC architektúra so smerovaním

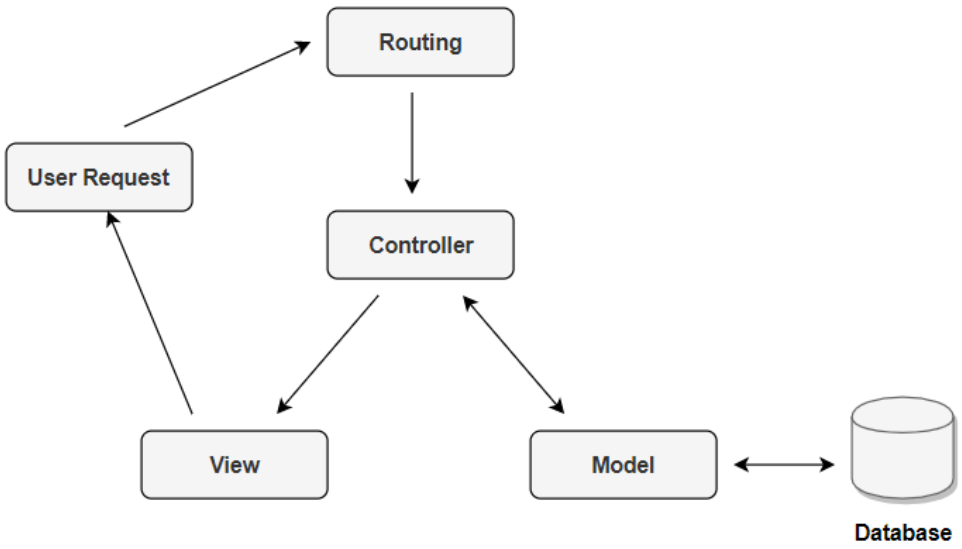
Zákaldnou funkciou každého frameworku webových aplikácii je prijímať požiadavky od používateľov a doručovať odpovede, zvyčajne prostredníctvom protokolu HTTP (s). To znamená, že definovanie smerovania aplikácie je jedna z dôležitých znalostí pri používaní

ktoréhokoľvek frameworku. Bez smerovania nie je takmer možné komunikovať s koncovým používateľom.

- Model - predstavuje individuálnu databázovú tabuľku (napr. zákazka alebo zákazník)
- View - predstavuje šablónu, ktorá sa odosiela koncovému používateľovi, s nastavenými HTML elementami, pridanými štýlmi a pripadne aj s Javascript kódom
- Controller - podobne ako dopravný policajt príjma požiadavky HTTP z prehliadača, získava správne údaje z databázy a ďalších mechanizmov úložiska, overuje vstup používateľa a nakoniec mu odošle odpoveď, controller môže v reagovať na túto požiadavku zapisovať údaje do modelu alebo načítať údaje z modelu (databázy), controller odošle údaje do view a tento pohľad sa potom zobrazí koncovému používateľovi

V aplikácii Laravel sa definuje smerovanie vo súbore web.php. Pri definovaní smerovania je potrebné určiť o aký HTTP príkaz ide (GET, POST, PUT a pod.), URL adresu, ktorá ho ma vyvolať a potom akciu, ktorá sa ma vykonať. Veľmi často sa ako akcia definuje konkrétna metóda controllera. Príklad, kde sa zavolá funkcia loginIndex z controlleru menom loginController po kliknutí na odkaz login:

```
Route::get('\login','loginController@loginIndex');
```




Obr. 5.1: MVC architektúra so smerovaním vo frameworku Laravel (zdroj: [13])

### Eloquent

Súčasťou Laravelu je Eloquent, ktorý je možné nazvať aj ako objektovo-relačný mapovač (angl. object-relational mapper). Slúži na objektovo-relačné mapovanie a umožňuje tak príjemnú komunikáciu s databázou. Pri použití Eloquent má každá databázová tabuľka

zodpovedajúci „Model, ktorý sa používa na interakciu s danou tabuľkou. Okrem načítania záznamov z databázovej tabuľky model Eloquent umožňujú vkladať, aktualizovať a mazať záznamy z tabuliek. [10]

### Artisan

Artisan je rozhranie príkazového riadku, ktoré je súčasťou Laravelu. Artisan existuje v koreňovom adresári aplikácie a poskytuje množstvo užitočných príkazov, ktoré môžu byť užitočné a pomôcť pri vytváraní aplikácie. Je možné, zobraziť všetky cesty v aplikácii, vytvoriť nový controller, model alebo vytvorenie iných tried s možnosťou predefinovať niektoré funkcie. [1]

# Kapitola 6

# Implementácia informačného systému

Z jednou najdôležitejších a najobšírnejších fázi celého vývoja informačných systémov je samotná implementácia. Popis implementácie v tejto kapitole vycháza z požiadavkov na systém a následného návrhu systému. V tejto kapitole budú teda rozobraté jednotlivé časti informačného systému. Na implementáciu boli použité technológie predstavené v kapitole 5. Pre prácu s databázou bol využívaný apache XAMPP, ktorý poskytol pri vývoji MySQL databázový systém.

Ako už bolo niekoľkokrát spomínané tak systém je rozdelený do šiestich častí (viď 3.3). To znamená, že klientska časť bola tiež rozdelená do šiestich šablón. Je možné ich nazvať aj vedľajšie šablóny. Vedľajšie preto, lebo týchto šesť vedľajších šablón ešte dopĺňa jedna hlavná šablóna, do ktorej sa vždy vloží obsah z práve jednej z týchto vedľajších šablón. Tým pádom bolo dosiahnuté, že elementy, ktoré sa zobrazujú na každej stránke (napr. menu) sa implementujú iba raz.

Štýl implementácie systému bol zvolený tak, že ak si užívateľ vyberie jednu zo šiestich časti systému, tak sa mu načíta HTML dokument a zvyšok interakcie so systémom v danej časti prebieha bez nutnosti kompletného znovunačitania (asynchrónna komunikácia). To je zabezpečené pomocou knižnice jQuery a technológie AJAX (Asynchronous JavaScript and XML).

Každá časť systému ma svoj controller, kde sa nachádzajú funkcie naprogramované v PHP. Tieto podprogramy v controllery buď pošlú celú šablónu s doplňujúcimi dátami z databázy alebo sú volané asynchrónne a tým pádom pošlú odpoveď klientskej časti v podobe vyžiadaných dát.

## 6.1 Pripojenie k databáze a práca s modelmi

Ako už bolo spomínané v kapitole 5.7 tak model predstavuje individuálnu databázovú tabuľku. Vo vyvíjanej webovej aplikácii je potrebné vytvorenie modelov, aby sme pohodlne mohlo pracovať s databázou, vkladať dáta do databázy alebo naopak načítať údaje, ktoré bude aplikácia potrebovať. Aby to všetko bolo možné využívať, je potrebné nakonfigurovať databázové služby framworku. Všetky nastavenia ohľadom databázy sa nachádzajú v konfiguračnom súbore *config / database.php*. V tomto súbore môžeme definovať všetky svoje databázové pripojenia a tiež určiť, ktoré pripojenie by sa malo predvolene použiť. Väčšina možností konfigurácie v tomto súbore sú založené na hodnotách v súbore *.env*. [5] Konkrétne

v prípade vyvíjanej aplikácie stačilo, nastavanie hodnôt, ktoré slúžia na pripojenie databázy k aplikácii. Na pripojenie databázy bolo potrebné nastaviť hodnotu *DB_CONNECTION* na *mysql*, *DB_HOST* menom mysql servera a hodnotu *DB_PORT* na číslo portu *3306*, čo predstavuje predvolený port pre používanie protokolu MySQL. Aby pripojenie k databáze fungovalo je ešte nevyhnutné vyplniť hodnotu *DB_DATABASE* menom databázy *DB_USERNAME* menom užívateľa databázy a prípadne *DB_PASSWORD* heslom. Nakoniec práce pri úprave súbora .env bol ešte upravený atribút *APP_NAME*, ktorý predstavuje meno aplikácie.

Po pripojení k databáze bolo potrebné vytvoriť databázové tabuľky. Na databázovom serveri bol použitý phpMyAdmin, čo je bezplatný softvérový nástroj napísaný v PHP, určený na správu MySQL cez web. [3] Následne boli vytvorené modely pre každú tabuľku. Na to, aby sa vedelo, ktorá databázová tabuľka zodpovedá, ktorému modelu bolo nevyhnutné dodržať určitú konvenciu. Táto konvecia spočíva v tom, že framework Laravel takýto problém vie riešiť sám, ale len v prípade, ak sa použije anglický názov tabuľky, ako množné číslo a názov modelu bude jednotné číslo názvu tabuľky (napr. názov tabuľky je posts a názov modelu post). Pri tvorbe databázových tabuliek boli použité slovenské názvy a teda databázové tabuľky nezodpovedajú tejto konvencií, tak bolo potrebné ručne určiť názov tabuľky modelu definovaním premennej typu `protected`.

## 6.2 Tvorba nových zákaziek

Jedna z požiadavok pri tvorbe nových zákaziek bola, aby sa pri vyplnaní údajov nestrácal zbytočný čas. Táto požiadavka sa dá splniť pri zadavní informácií o zákazníkovi a farbe (viď 3.3). Ak užívateľ systému začne vyplňovať meno zákazníka, tak klientska časť požiada server o zákazníkov v systéme, ktorí zadanú časť mena spĺňajú. Udalosť zaslania požiadavkou vyvolá metóda knižnice jQuery `.keyup()`, ktorá zasiela požiadavok po každom uvoľnení klavesnice. Server vráti zoznam zákazníkov a po vybratí príslušneho zákazníka užívateľom sa vyplnia všetky údaje o zákazníkovi (e-mail, tel. číslo, ičo a pod.). Podobne sa systém správa pri vyplnení údajov o farbe.

Pri tvorbe nových zákaziek užívateľa sprevádza niekoľko modálnych okien, ktoré upozorňujú užívateľa a vyzývajú ho na odpoveď, ktorá špecifikuje, ako sa systém ma zachovať. Napríklad systém sa pýta či môže zákazku radiť za inú alebo upozorňuje, že tvorená zákazka nie úplná a bude potreba ju v najbližšom čase dokončiť. Modálne okná boli implementované pomocou Boostrap frameworku, ktorý poskytol HTML elementy, css pravidlá a nakoniec aj potrebný javascript. Pri implementácií stačilo zosúladiť (musia byť zhodné) atribút `data-target` tlačidla, ktoré modálne okno vyvolá, s identifikátorom HTML elementu, ktorý predstavuje dané modálne okno. Aby sme boli úplne presný tak tlačidlo, ktoré vyvolá modálne okno nestlačí užívateľ, ale vo väčšine situácií stlačenie tlačidla vyvolá metóda `.trigger('click')` ešte pred zaslaním dát na server. Túto metódu ponúka opäť knižnica jQuery. Týmto prístupom si systém vyžiada interakciu s užívateľom kedy on potrebuje a nedôjde k nekonzistencií systému.

## 6.3 Radenie zákazok

Radenie zakázok spočíva v zmene atribútu poradie (určuje poradie zákazky v pláne) v entitnej množine zákazka. Pri implementácií radenia zakázok sa musí myslieť nie len na zmenu atribútu radenej zákazky, ale aj zmenu atribútov zvyšných zákaziek, ktorých sa zmena

dotýka. Tento mechanizmus bol implementovaný nasledovne. Načítajú sa všetky zákazky v pláne výroby, ktoré sa postupne prechádzajú v cykle. Ak systém narazí na zákazku s rovnakou farbou, tak radenej zákazke sa pridelí poradie o jedno väčšie ako nájdenej zákazke. Ak sa jedná o skupinu zákaziek s rovnakou farbou, tak systém zaradí novú zákazku na koniec celej skupiny (viď 3.3). Ďalším krokom pri radení bolo zvýšenie poradia o jedno všetkým zákazkám, ktoré sa nachádzajú za novou už zaradenou zákazkou. Môže nastať aj situácia, že zákazka nebude vkladaná do plánu, ale bude sa meniť len jej miesto v pláne. V takom prípade, je potrebné, aby podprogram, ktorý radenie vykonáva si pamätal pôvodné poradie zákazky a prispôsobil ostatné hodnoty iných zákaziek. V podstate vždy dochádza k prechádzaniu celého plánu a zvyšovania respektíve znižovania hodnoty atribútu poradie v entitnej množine zákazka.

## 6.4 Práca s plánom výroby

Jedna z najuzaujímavejších funkcionalít pri práci a s plánom výroby je zmena poradia zákaziek v pláne. Hlavná myšlienka pri implementácii tohto požiadavku bola vytvoriť "drag and drop"metódu. Knižnica jQuery ponúka aj sadu nástrojov pre lepšie užívateľské rozhrania. Jednou z nich je nástroj `Sortable`, ktorý povolí triediť skupinu prvkov HTML dokumentu. Potom už stačí len kliknúť na prvok a presunúť ho na nové miesto v zozname. Ostatné položky sa prispôsobia novému rozmiestneniu. [15] Po novom poradí zákaziek sa pošle asynchornne požiadavok na server aj s novým poradím, ktoré sa uloží do databázy.

Pri implementácií plánu sa myslelo aj na problém, ktorý by mohol vzniknúť pri prehodení poradia zakázok v kancelárií. Môže nastať totiž problém, že ak by výroba mala načítaný plán a nedošlo by k aktualizácii nového poradia v pláne, tak výroba by mohla pracovať so starými informáciami. Preto sa klientska časť dotazuje každých päť sekúnd či nedošlo k zmene. Ak áno, tak plán sa aktualizuje. Podobným spôsobom dochádza aj k aktualizácii ak nastanú nové informácie o požiadavkách zákazníka alebo nastane zmena pri množstve farby na sklade. Informácia o nedostatku farby zmizne a zamestnanci vo výrobe sa môžu pustiť do prípravy novej zákazky. Rovnakým spôsobom funguje aj zmena stavu zákazky, aby naopak kancelária nestratila prehľad o aktuálnom stave výroby.

V pláne sú implementované modálne okná rovnakým spôsobom, ako pri tvorbe novej zákazky (viď 6.2).

## 6.5 Údaje o skladových zásobách

Pre lakovaciu firmu je kľúčové, aby mala stručný prehľad o svojich skladových zásobách (viď 3.1). Na tento prehľad sa pri implementácií využila tabuľková forma s podporou doplnku DataTables pre javascriptovú knižnicu jQuery. Bolo potrebné importovať pár skriptov pre správne fungovanie tabuľky spolu s nastaveniami tabuľky v slovenskom jazyku, ktorý už vytvorili iní autori [7].

Pre naplnenie tabuľky bola využitá sila frameworku, kedy je možné spolu so šablonou poslať všetky dáta, ktoré súvisia so skladovými zásobami. Väčší problém už bolo zobrazenie štatistík v podobe grafov tak, aby si ich užívateľ mohol pohodlne zobraziť a nestratiť zároveň prehľad o stave iných farbách. Preto bola implementovaná funkcionalita vytvorenia nového priestoru s grafom a inými informáciami o farbe hneď pod riadok v tabuľke s danou farbou. Po kliku na riadok s farbou sa teda hneď pod ním zobrazí graf. Na to, aby táto myšlienka fungovala sa musí vytvoriť úplne nový element a vložiť priamo do dokumentu HTML na

správne miesto. Tentokrát sa využila metóda `.after('html element')` opäť od knižnice jQuery. Táto metóda vloží obsah jeho argumentu za iný špecikovaný element. Ďalším krokom bolo už len implementovať zobrazenie grafov. Použitie Chart.js nie je náročný proces. Vyžaduje sa iba zahrnutie skriptu na stránke spolu s jediným elementom `<canvas>`. Aby graf mohol byť vytvorený, musí sa vytvoriť inštancia triedy Chart a odovzdať jej už spomenutý element canvas.

## 6.6 Prihlásenie uživateľa

Systém potrebuje vedieť, ktoré časti informačného systému môže prihlásenému užívateľovi zobraziť. Na to boli využité tzv. `sessions`. Pretože aplikácie riadené protokolom HTTP sú bezstavové, `session` poskytuje spôsob ukladania informácií o užívateľovi na strane servera. Ak sa tedá uživateľ prihlási pod prihlasovacím menom, ktoré má v databáze práva výroby, tak systém mu neumožní vidieť iné užívateľské rozhrania, ako je plán výroby. Po odhlásení užívateľa sa pochopiteľne táto informácia na strane servera vymaže.

## 6.7 Konvertovanie farieb do RGB modelu

Lakovacia firma pracuje s farbami, ktoré majú odtieň špecifikovaný vzorkovníkom RAL (viď 4.1). Jeden z cieľov pri implementácií klietskej časti bol navodiť užívateľovi, pri práci s farbami, väčší prehľad s čím pracuje. Preto takmer na každom mieste, kde sa nachádza informácia o odtieni farby, tak tento odtieň bol prekonvertovaný v užívateľskom rozhraní do RGB modelu. Túto konverziu v systéme zabezpečuje tzv. `simple color converter`, ktorý pokrýva väčšinu farebných formátov vrátane RAL a je schopný vrátiť farbu vo formáte RGB modelu.[14] Bolo potrebné nainštalovať tento konvertér jedným príkazom v príkazovej riadke pomocou npm, ktorý je správcom balíkov pre platformu Node JavaScript.

## 6.8 Export dát

Jedna z požiadavok na informačný systém bola aj možnosť exportovať dáta von zo systému či už do papierovej podoby alebo do iných formátov (napr. PDF, XLSX, alebo CSV). Na splnenie tohoto požiadavku sa skvelé hodí rozšírenie od doplnku Datatables, ktorý je súčasťou knižnice jQuery. Rozšírenie tzv. "buttons", poskytuje doplnky, ktoré umožňujú funkciu exportu údajov. Využíva sa `API5 HTML5` na vytváranie súborov na strane klienta a program `Adobe Flash` pre staršie prehliadače. [6] Ako to býva u datatables zvykom, tak na prídanie konkrétneho rozšírenia do už vytvorenej tabuľky stačí pridať do konštruktora dátových tabuliek vhodné nastavenie podľa dokumentácie. Po dokončení konfigurácie nad tabuľkou sa objavia tlačidlá určujúce formát exportu a tiež automaticky určia, či by sa mal používať formát `HTML5` alebo `Flash` na základe funkčnosti prehliadača.

K exportu dát sa ešte hodí spomenúť riešenie, ktoré umožňuje exportovať vytvorený graf do formátu PNG. Na to bola použitá metóda `.toDataURL()`, ktorá vracia dáta URI obsahujúce reprezentáciu obrázka vo formáte určenom v parametri (predvolene PNG).

# Kapitola 7

# Testovanie

Testovanie softvéru je veľmi dôležitá fáza celého vývoja. Je takmer kľúčové skontrolovať všetko, čo bolo vytvorené, pretože programované časti sa vždy môžu preukázať nejakou chybou. Nehovoriac o tom, že sme ľudia a robíme chyby. Keďže každý človek robí občas chyby, je nevyhnutné si vždy po sebe skontrolovať svoju prácu. V najlepšom prípade by mohlo byť ideálne požiadať niekoho iného, aby skontroloval vytvorený systém, pretože často úplne iný človek nájde s vyššou pravdepodobnosťou nedostatky a chyby, ktoré by si autor práce nemusel vôbec všimnúť. Takto tomu bolo aj v tomto projekte. Počas procesu vývoja informačného systému, po naprogramovaní určitej časti, bola snaha aspoň na nižšej úrovni testovania ručne odskúšať funkcionalitu novej časti systému, alebo častí systému, ktoré svojim spôsobom na ňu nadväzovali.

Systém si skúšal pravidelne majiteľ firmy a zároveň doplňoval požiadavky. Tento spôsob odhalil nedostatok v organizácií projektu, pretože, ak zákazník uvidí na vlastne oči a vyskúša si časť systému, tak mu napadne hneď niekoľko inovácií. Doplňujúce požiadavky možno zlepšovalo celkovú kvalitu systému z pohľadu zákazníka, ale veľmi zdržovalo vývoj. Preto je lepšie vytvoriť na začiatku projektu prototyp, ktorý si zákazník vyskúša, napíšu sa nové požiadavky a potom sa vytvorí finálny produkt. Testovanie má mnoho ďalších výhod a jednou z nich je časová efektívnosť. Testovanie v tomto projekte v pár prípadoch ušetrilo čas na opravy, lebo chyba bola odhalená v čas. Vývoj softvéru pozostáva z mnohých fáz, a ak sú chyby zachytené v skorších fázach, tak sú časové náklady na ich opravu omnoho nižšie. Preto je dôležité, aby sa testovanie vykonávalo, čo najskôr.

Pri testovaní počas celého vývoja, prešlo systémom 403 zakázok, 121 farieb a 19 zákazníkov. Zamestnancov bolo vytvorených 10. Testovanie odhalilo pár desiatok chýb vo funkcinalite a niekoľko nezrovnalostí v požiadavkách na systém. Hromadnejšie testovanie na konci vývoja ukázalo pomerne vyššiu neintuitívnosť užívateľského rozhrania. Nemusí to byť však obrovský problém, keďže so systémom budú pracovať rovnakí ľudia a bude ich pomerne malý počet. To znamená, že na systém sa dá zvyknúť, čo potvrdili aj respondenti, ktorí systém testovali. Neintuitívnosť rozhrania však porušuje princípy dobrej webovej aplikácie a ako aj testovanie ukázalo pre príjemnejšie používanie bude potrebné pár častí informačného systému zjednodušiť. Testovanie však tiež ukázalo, že ak si užívateľ naštuduje pripravený návod na používanie systému alebo prípadne mu je vysvetlená funkcionalita, tak pre užívateľa sa systém stáva viac intuitívny.

# Kapitola 8

# Záver

Cieľom tejto bakalárskej práce bolo navrhnúť a implementovať informačný systém firmy zaoberajúcej sa povrchovými úpravami. Pre lepšie porozumenie problematiky povrchovej úpravy bolo potrebné nielen podrobne preštudovať techniku práškového lakovania ale zoznámiť sa aj s fungovaním celého výrobného procesu firmy. Navrhnuté nové riešenia v podobe časti informačného systému boli navrhnuté autorom práce v spolupráci s majiteľom lakovacej firmy, ktorý postupne upresňoval svoje požiadavky. Celý proces vývoja bol naročný práve v tom, že zákaznícková predstava sa neustále vyvíjala spolu so systémom.

V práci boli spomenuté aj technológie, s ktorými autor prišiel počas vývoja do kontaktu. Špeciálna pozornosť určite patrí frameworku Laravel, ktorý celý vývoj zjednodušil. Bolo si potrebné naštudovať praktiky pre vývoj webových aplikácií a hlavne zoznámiť sa s problematikou práve PHP frameworku Laravel. Nasledujúcimi kapitolami už boli praktické etapy vývoja informačného systému. Výsledkom analýzy požiadavkov bol diagram prípadov použitia, v ktorom boli uvedené základné funkčné požiadavky systému. Bol spomenutý aj návrh informačného systému, kde pomocou ER diagram boli predstavené dáta v systéme spolu s jednotlivými entitnými množinami a vzťahmi medzi nimi. Následne bolo stručné predstavené grafické užívateľské rozhranie. Následne sa čitateľ môže dozvedieť viac o implementácii, v ktorej boli popísané základné časti systému spolu s testovaním.

Na koniec projektu sa dá skonštatovať, že zadanie bakalárskej práce bolo splnené. Informačný systém je momentálne vo fáze dlhodobejšieho testovania, aby mohol byť následne nasadený do praxe. Kvôli neustále pridavajúcim požiadavkám od zákazníka, ktoré predstavujú viacmenej detaily, občas väčšiu pridanú časť bol zvolený nasledovný postup. Vyhodnotiť dlhodobejšie testovanie a upraviť aplikáciu o nové požiadavky od zákazníka.

## 8.1 Čo by sa dalo zdokonaliť

V práci je rozobraté riešenie zoskupenia zákaziek rovnakej farby nachádzajúce sa v pláne. Testovanie ukázalo, že pre lakovaciu firmu, ktorá bola tejto práce súčasťou, je momentálne riešenie postačujúce a veľmi oceňuje možnosť manuálnej rýchlej zmeny poradia zákaziek v pláne. Slová firmy po testovaní plánu sú, že zhodnotenie komplexnosti zákazky je veľmi individuálne a závisí od množstva faktorov. Fungovanie firmy nepredstavuje veľkovýrobu s rovnakými pravidelnými objednávkami, ale ponúka individuálny prístup ku každej objednávke. Riešenie na základe upozornenia rovnakej farby v pláne a následné zoskupenie je podľa slov majiteľa firmy postačujúce. Napriek tomu by sa riešenie dalo rozšíriť o nejaké ohodnotenie komplexnosti zákazky a podľa tohto ohodnotenia by systém mohol zoskupovať

väčšie zákazky s menšími. Tiež by sa do plánovania dali zakomponovať aj dátumy. Takéto riešenie by však bolo veľmi závislé na testovaní a až nasadenie do praxe by ukázalo, ako je tento návrh efektívny.

Cieľ pri zadavní zákaziek bol, čo naviac zrýchliť a zefektívniť proces špecifikácie novej objednávky. Riešenie pozostáva z toho, že užívateľ vyplňuje vstupy a tie mu ponúkajú možnosti dát, ktoré sa v systéme už nachádzajú. Testovanie ukázalo, že užívateľ by možno viac ocenil len variantu výberu možnosti z databázy a v prípade novej hodnoty atribúty, by systém obsahoval aj variantu rýchleho pridania novej hodnoty. Užívateľ by tak vedel "naklikať"zákazku prakticky len pomocou myši a nepotreboval by v niektorých prípadoch ani klávesnicu. Postupne by systém bol obohatený o väčšie množstvo hodnôt a užívateľ by eleminoval prístup rúk na klávesnicu. Pri testovaní zadavania užívatelia stískali enter s vedomím, že ich to posunie na ďalší vstup, iný však pužívali jednou rukou myš. Je to individuálny problém ale ako ukázalo testovanie vo väčšine je preferovaná vyššie spomuntá možnosť len "naklikania".

Systém obsahuje exportovanie dát v podobe tabuliek a ich vyflitrovaných riadkov. Je tu plne na mestie rozšíriť funkcionalitu o export len niektorých stĺpcov tabuliek. Stále je však možné stĺpce odstrániť mimo systému (napr. pri exportovaní tabuľky do programu Excel).

# Literatúra


[1] *Artisan Console - Laravel - The PHP Framework For Web Artisans* [online]. [cit. 2021-22-04]. Dostupné z: https://laravel.com/docs/8.x/artisan.

[2] *Bootstrap - The most popular HTML, CSS, and JS library in the world* [online]. [cit. 2021-02-04]. Dostupné z: https://getbootstrap.com/.

[3] *Bringing MySQL to the web* [online]. [cit. 2021-20-04]. Dostupné z: https://www.phpmyadmin.net/.

[4] *Chart.js | Open source HTML5 Charts for your website* [online]. [cit. 2021-20-04]. Dostupné z: https://www.chartjs.org/.

[5] *Database: Getting Started - Laravel - The PHP Framework For Web Artisans* [online]. [cit. 2021-20-04]. Dostupné z: https://laravel.com/docs/8.x/database.

[6] *DataTables example - File export* [online]. [cit. 2021-21-04]. Dostupné z: https://datatables.net/extensions/buttons/examples/initialisation/export.html.

[7] *DataTables in Slovak* [online]. [cit. 2021-20-04]. Dostupné z: https://datatables.net/plug-ins/i18n/Slovak.

[8] *Datatables manual* [online]. [cit. 2021-20-04]. Dostupné z: https://datatables.net/manual/index.

[9] *Electrostatic Powder Coating VS Fluidized Bed Powder Coating* [online]. [cit. 2021-20-03]. Dostupné z: https://www.colopowdercoating.com/Electrostatic-Powder-Coating-VS-Fluidized-Bed-Powder-Coating-id3222464.html.

[10] *Eloquent: Getting Started - Laravel - The PHP Framework For Web Artisans* [online]. [cit. 2021-30-04]. Dostupné z: https://laravel.com/docs/8.x/eloquent#introduction.

[11] *JQuery API documentation* [online]. [cit. 2021-04-04]. Dostupné z: https://api.jquery.com/.

[12] *JQuery Introduction* [online]. [cit. 2021-02-04]. Dostupné z: https://www.w3schools.com/jquery/jquery_intro.asp.

[13] *Laravel interview questions* [online]. [cit. 2021-20-04]. Dostupné z: https://interview-guru.club/laravel.

[14] *Simple color converter* [online]. [cit. 2021-20-04]. Dostupné z: https://www.npmjs.com/package/simple-color-converter.

[15] *Sortable | jQuery UI* [online]. [cit. 2021-04-04]. Dostupné z: https://jqueryui.com/sortable/.

[16] *Why Laravel is the Best PHP Framework to Use* [online]. [cit. 2021-25-04]. Dostupné z: https://www.freecodecamp.org/news/why-laravel-is-the-best-php-framework-to-use-in-2020/.

[17] Gilmore, W. J. *Beginning PHP and MySQL: From Novice to Professional.* 4. vyd. Paul Manning, 2010. ISBN 978-1-4302-3115-8.

[18] Křena, B. a Kočí, R. *Úvod do softwarového inženýrství IUS Studijní opora.* FIT VUT v Brně, 2010.

[19] Makhlouf, A. S. H. a Abu Thabit, N. Y. *Advances in smart coatings for magnesium alloys and their applications in industry.* 1. vyd. Elsevier Inc., 2019. ISBN 978-0-12-849870-5.

[20] Stauffer, M. *Laravel: UP & Running.* 2. vyd. O'Reilly Media Inc., 2019. ISBN 978-1-492-04121-4.

[21] Tomaško, T. *ARIES - práškova lakovňa* [online]. [cit. 2021-20-03]. Dostupné z: http://www.komaxit.sk/.

[22] Welling, L. a thomson, L. *Mistrovství PHP a MySQL.* 1. vyd. Pearson Education, Inc, publishing as Addison-Wesley Professional, 2017. ISBN 978-80-257-4892-1.

[23] Wempen, F. *HTML5 Step by Step.* Octal Publishing, Inc., 2011. ISBN 978-0-735-64526-4.

[24] Žára, O. *JavaScript, Programatorské techniky a webové technologie.* 1. vyd. Computer Press v spoločnosti Albatros Media a.s., 2015. ISBN 978-80-251-4573-9.

# Príloha A

# Obsah pamäťového média

Pamäťové médium obsahuje adresár **laravel**, kde sa nachádzajú zdrojové súbory informačného systému. V adresári **db** je umiestnený súbor, ktorý obsahuje skript na vygenerovanie a naplnenie ukážkovej databázy. V adresári **doc** sa nachádza táto technická správu vo formáte PDF a zdrojové súbory, pomocou ktorých je možné túto technickú správu vygenerovať. Pamäťové médium ešte obsahuje textový súbor *manual.pdf*, ktorý predstavuje manuál na používanie systému a *README.md*, kde je popísaný hlavne postup pri inštalácií systému.