Mendelova univerzita v Brně
Provozně ekonomická fakulta

E-shop s možnosťou užívateľskej konfigurácie produktov

Bakalárska práca

Vedúci práce:
Ing. Jiří Lýsek, Ph.D.

Žaneta Kučarová

Brno 2017

Rada by som poďakovala Ing. Jiřímu Lýskovi, Ph.D. za jeho cenné rady a čas, ktorý mi venoval.

Čestné prehlásenie

Prehlasujem, že som túto prácu: **E-shop s možnosťou užívateľskej konfigurácie produktov**
vypracovala samostatne a všetky použité pramene a informácie sú uvedené v zozname použitej literatúry. Suhlasím, aby moja práca bola zverejnená v súlade s § 47b zákona č. 111/1998 Sb., o vysokých školách v znení neskorších predpisov, a v súlade s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*
Som si vedomá, že sa na moju prácu vzťahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzatvorenie licenčnej zmluvy a použitie tejto práce ako školského diela § 60 odst. 1 Autorského zákona.
Ďalej sa zaväzujem, že pred spísaním licenčnej zmluvy o využití diela inou osobou (subjektom) si vyžiadam písemné stanovisko univerzity o tom, že predmetná licenčná zmluva nie je v rozpore s oprávenými záujmami univerzity, a zaväzujem sa uhradiť prípadný príspevok na úhradu nákladov spojených s vznikom diela, a to až do ich skutočnej výšky.

Brno, 1.1.2017 .....

Abstract

KUČAROVÁ, Ž. *E-shop solution for the user's product customization.* Bachelor thesis. Brno, 2017

Bachelor thesis is focusing on an implementation of the e-shop solution for the user product configuration. This should bring competitive advantage to the submitter and higher added value to the consumer when shopping.

In the first part, thesis deals with a product customization, description of the customization process and shows an extended model for the product customization. Further it describes the company Helena Išová, for whom the solution is created. Described is also an overview of existing solutions, which address product customization across different e-commerce stores. At the end of this part, thesis describes the development of modern web applications and dedicated resources.

In the methodology part, thesis describes the CodeIgniter framework which uses MVC architecture. This will be used for the implementation of proposed solution. Other software tools described are also UML diagrams, MySQL, Javascript and its library.

The last chapter consists of results and can be divided into two parts. First describes the proposed solution with the corresponding administrative and customer interface, as well as the implementation process. The proposed implementation process includes the use case diagram based on the functional and non-functional requirements. It also includes an entity relationship model for the structure of data saving and wireframes for the user interface design. The second part of results describes the solution for the saving and color changes in the configuration application, the procedure of obtaining data from the server using Ajax and in the end the focus is on the implementation of a custom templating system.

Keywords

product customization, modern web application development, MVC architecture, CodeIgniter, PHP

Abstrakt

KUČAROVÁ, Ž. *E-shop s možnosťou užívateľskej konfigurácie produktov.* Bakalárska práca. Brno, 2017

Bakalárska práca sa zaoberá implementáciou e-shopu s riešením užívateľskej konfigurácie produktov, ktorá má priniesť zadávateľovi konkurenčnú výhodu a jeho klientom zvýšiť pridanú hodnotu pri nákupe.

V prvej časti sa práca venuje definícii produktovej kustomizácie, popisu priebehu kustomizácie a uvádza rozšírený model kustomizácie produktu. Ďalej predstavuje zadávateľa Helenu Išovú, pre ktorú je riešenie vyvíjané. Uvedený je taktiež prehľad existujúcich riešení, ktoré sa venujú riešeniu produktovej kustomizácie v internetových obchodoch. Na záver tejto časti je popísaný vývoj moderných webových aplikácií a prostriedkov k tomu určených.

V Materiáloch a metodike je opísaný framework CodeIgniter používajúci MVC architektúru, ktorý bude využitý pri implementácii riešenia, a spomenuté sú aj ďalšie programové nástroje ako UML diagramy, MySQL, Javascript a jeho knižnice.

Kapitolu Výsledky môžeme rozdeliť na dve časti, a to návrh riešenia so zodpovedajúcim administračným a zákazníckym rozhraním a popis samotnej implementácie. V návrhu implementácie je na základe funkčných a nefunkčných požiadaviek zostrojený diagram prípadov použitia. Neskôr je pripravený entitno-relačný model pre štruktúru ukladania dát v aplikácii a drôtové modely pre návrh užívateľského rozhrania. V druhej časti kapitoly je podrobne popísané riešenie spôsobu ukladania a zmeny farieb v konfiguračnej aplikácii, postup získavania dát zo serveru pomocou Ajaxu a nakoniec je venovaná pozornosť implementácii vlastného šablónovacieho systému.

Kľúčové slová

produktová kustomizácia, vývoj moderných webových aplikácií, MVC architektúra, CodeIgniter, PHP

Obsah

Zoznam obrázkov

1 Úvod a cieľ práce

1.1 Úvod

Rastúca konkurencia a pokles vernosti zákazníkov vzhľadom k značkám je problémom takmer všetkých firiem naprieč rôznymi odvetviami. Technológie a internet umožňujú zákazníkom jednoducho porovnávať ponúkané produkty a dosahovať aj na predtým neprístupné trhy. Firmy sú preto nútené investovať do inovácie, vývoja nových alebo špeciálnych produktov s pridanou hodnotou, ktorú konkurencia nie je schopná ponúknuť. Otázkou je, ako takúto pridanú hodnotu získať a poskytnúť zákazníkovi. Jedným z riešení je zavedenie možnosti produktovej kustomizácie, ktorá môže priniesť konkurenčnú výhodu a zvýšiť vernosť zákazníkov vzhľadom ku značke, nakoľko umožňuje väčšie uspokojenie potrieb zákazníka.

Predstavy zákazníka sú vyjadrované formou špecifikácií, ktoré je potrebné presne definovať, aby neprišlo k nepochopeniu zo strany výrobcu. Zadávateľka sa často stretávala s otázkou, ako čo najlepšie zachytiť požiadavky zákazníkov, a zároveň podporiť ich predstavivosť. V minulosti riešila špeciálne požiadavky na dizajn kabeliek e-mailovou komunikáciou alebo osobným stretnutím, čo však bolo často časovo neefektívne. Práve tento problém má riešiť vizualizácia špecifikácií v rámci konfiguračnej aplikácie. Navyše má zlepšovať užívateľskú skúsenosť pri nákupe, pretože sa požadované zmeny zákazníkovi zobrazia ihneď vo voľbe. Možnosť produktovej kustomizácie v internetovom obchode medzi konkurenčnými značkami klientky nie je zvyčajná, preto môže predstavovať významnú konkurenčnú výhodu.

Tieto fakty viedli klientku k rozhodnutiu umožniť svojim zákazníkom produkty upravovať cez konfiguračnú aplikáciu v rámci internetového obchodu.

1.2 Cieľ práce

Cieľom tejto bakalárskej práce je vytvorenie e-shopu s konfiguračnou aplikáciou, ktorá umožní užívateľskú kustomizáciu produktu v rámci e-shopu. Pre dosiahnutie tohto cieľa je potrebné špecifikovať funkčné a nefunkčné požiadavky, preskúmať existujúce riešenia na trhu a na základe zistení rozhodnúť či využiť existujúce riešenie alebo vlastnú implementáciu. V závere práce budú vyhodnotené výsledky a budú navrhnuté ďalšie zlepšenia a rozšírenia súčasného stavu.

2 Prehľad literatúry

2.1 Kustomizácia produktu

Pine a Gilmore (1999) definujú kustomizáciu produktu[1] ako „...vyrábanie na základe špecifických požiadaviek zákazníka.“ Kustomizácia má prinášať zákazníkovi viditeľnú pridanú hodnotu, nakoľko v porovnaní s produktom z masovej výroby kustomizovaný produkt uspokojuje potreby zákazníka v oveľa väčšej miere (Blecker a kol., 2004). To je spôsobené kladeným dôrazom na individuálne požiadavky zákazníkov pri konkrétnych spôsoboch výroby. Pri produkte z masovej výroby je dôležitý dizajn a požiadavky všeobecného profilu zákazníka, avšak pri kustomizovaných produktoch sa postupuje podľa špecifických požiadavok konkrétneho zákazníka.

Niektoré z predpokladov na použitie kustomizácie sú:

1. Špecializovaná marketingová stratégia s pevne definovanou úzkou cieľovou skupinou a s ňou spojeným sortimentom, resp. skupinou produktov, ktorá bude cieľovej skupine ponúkaná.
2. Definícia výrobných špecifikácií pre kustomizované produkty je akceptovaná a realizovateľná vo výrobe.
3. Sortiment založený na moduloch, kde výsledný produkt je výsledkom výberu, kombinácie a úpravy štandardných modulov (Hvam, Mortensen, Riis, 2008).

Spaulding a Perry vyzdvihujú základné body, ktoré sú potrebné pri úspešnej aplikácii produktovej kustomizácie. Špecifikácia vlastných očakávaní organizácie pred zavedením kustomizácie slúži k správnemu nastaveniu ďalšej stratégie v rámci organizácie. Napríklad v prípade, že firma očakáva nárast vernosti zákazníkov k značke, náklady sú považované za súčasť marketingových výdajov. Ďalším dôležitým bodom je stanovenie rozsahu možnosti kustomizácie. Produkt môže byť kustomizovaný pridaním ďalšej funkcionality či vlastného dizajnu už existujúcemu variantu produktu alebo kompletným zostavením nového variantu produktu z dostupných modulov. Proces kustomizácie však musí byť jednoduchý a prehľadný, inak môže množstvo možností potenciálnych zákazníkov zahltiť a odradiť od nákupu (Spaulding, Perry, 2013).

2.1.1 Konkurenčná výhoda

Firmy v dnešných dňoch čelia rastúcej konkurencii a dopyte po špecializovaných produktoch. Práve použitie možnosti kustomizácie vlastného produktu im môže priniesť obrovskú konkurenčnú výhodu na trhu. Pridaná hodnota je vnímaná v momente, keď sú za ňu ochotní zákazníci zaplatiť. Môže byť nadobudnutá ponukou rovnakých benefitov za nižšie ceny ako konkurencia alebo unikátnymi benefitmi, za ktoré môžeme žiadať vyššiu cenu (Porter, 1995). Ako príklad unikátneho benefitu

[1] Z anglického výrazu Product customization.

môžeme uviesť špecifický dizajn, kvalitné prevedenie alebo samotnú značku produktu, ktoré sú najčastejšie vnímané ako pridaná hodnota pre dizajnérske produkty.

Kustomizácia posilňuje vernosť zákazníka vzhľadom ku značke, čo je prínosné vzhľadom ku súčasným trendom. V nedávnom výskume od firmy Bain (2013), v ktorom sa zúčastnilo 1200 výkonných riaditeľov z rôznych oborov a krajín, až 67 % zúčastnených uviedlo, že ich firma pociťuje pokles vernosti zákazníkov vzhľadom k ich značke. Technológie a internet umožňujú užívateľom detailné porovnávanie cien a funkcií produktov a taktiež uľahčujú porovnávanie a vyhľadávanie produktov s klasickými funkciami. Diferenciácia kustomizovaných produktov a ponuka špeciálnych funkcií či úprav sa v tomto prípade považujú za spôsob ako zaujať potenciálneho zákazníka a priviesť ho do internetového obchodu.

Ďalšou nepochybnou výhodou kustomizácie je široká databáza nápadov dizajnu produktu priamo od zákazníkov. Táto skutočnosť môže byť využitá ako spätná väzba od zákazníkov a zároveň ako odrazový mostík pri dizajne nových funkcií či modelov produktov (Spaulding, Perry, 2013).

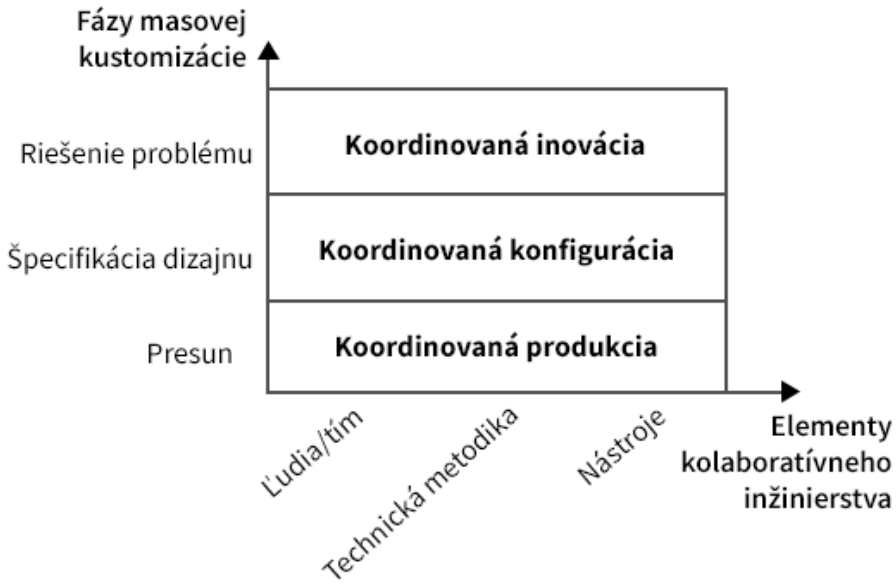
2.2 Proces kustomizácie produktu

Základný model procesu kustomizácie produktu definoval Spring a kol. (2000). Model pozostáva z troch častí – Riešenie problému a inovácia, Špecifikácia dizajnu a konfigurácia, a nakoniec Transformácia a výroba. Jeho nedostatkom vzhľadom ku kustomizácii produktu je chýbajúca možnosť komunikácie medzi zákazníkom a výrobcom pri špecifikácii dizajnu kustomizovaného produktu.

Chen, Wang a kol. (2009) navrhli rozšírený model procesu kustomizácie produktu, ktorý umožňuje integráciu zákazníka nielen do riešenia problému, ale aj do procesu špecifikácie produktu a samotnej výroby. V porovnaní s predošlým modelom sú jednotlivé fázy pomenované ako koordinovaná inovácia[2], koordinovaná konfigurácia[3],a na záver koordinovaná produkcia[4]. Tieto sú vložené do modelu s ďalšími troma hlavnými determinantami a to ľudia/tím, technická metodika a nástroje v rámci perspektívy kolaboratívneho inžinierstva[5].

Neustále zmeny požiadaviek zákazníkov zapríčiňujú potrebu hľadania nových riešení alebo inovácie už existujúcich riešení. Koordinovaná inovácia zastrešuje procesy potrebné k získaniu ideálneho riešenia pre obe strany, ako konečných zákazníkov, tak výrobcov. Koordinovaná produkcia zahrňuje prípravu a prepravu materiálov, management skladov a procesy spojené s výrobou produktu. Spolupráca s dodávateľmi a konečnými zákazníkmi prináša ďalší nárast výkonnosti produkcie, zvýšenie zodpovednosti a pokles nákladov. Viac informácií ohľadne každej fázy je možné nájsť v článku Mass Customization as a Collaborative Engineering Effort od

[2]Voľne preložené z anglického co-innovation.
[3]Voľne preložené z anglického co-configuration.
[4]Voľne preložené z anglického co-production.
[5]Voľne preložené z anglického collaborative engineering perspectives.



Obrázok 1: Rozšírený model kustomizácie produktu (Chen, Wang, Tseng, 2009)

Chen, Wang a kol. (2009). Pre potreby tejto práce bude podrobnejšie preberaná iba fáza koordinovanej konfigurácie.

2.2.1 Koordinovaná konfigurácia

Fáza, kde sa stretáva zákazník s výrobcom s cieľom definovania špecifikácií kustomizovaného produktu, sa nazýva konfigurácia. Jedná sa o proces výberu existujúcich alternatív tvoriacich špecifický variant produktu, ktorý uspokojí ako zákazníka, tak i výrobcu. Kvalita procesu konfigurácie je pre spokojnosť zákazníka kľúčová, pretože priamo ovplyvňuje mieru uspokojenia jeho individuálnych potrieb. Správna konfigurácia je však náročný proces, nakoľko vyžaduje správne pochopenie potrieb zákazníka a možností výrobcu a ich následnú efektívnu kombináciu. V tejto fáze je nemenej dôležitá správna interpretácia a zaznamenanie vybraných špecifikácií, ktoré zostavujú výsledný produkt.

Technológie sa ponúkajú ako ideálny prostriedok na zachytenie špecifikácií zákazníka, nakoľko dokážu umožniť flexibilné, digitálne kontrolované výrobné procesy a integráciu výroby a online dizajnu. Firmy tak jednoduchšie poskytujú zákazníkovi možnosť online dizajnu s príjemnou užívateľskou skúsenosťou. Navyše krátky čas odpovede systému na požiadavok zákazníka vo veľkej miere zlepšuje kvalitu procesu kustomizácie (Chen, Wang, Tseng, 2009).

2.3 Špecifikácie od zákazníka

Za účelom vyjadrenia požiadaviek zákazníka je potrebné definovať špecifikácie, ktoré vyjadria detailnú podobu výsledného produktu. Špecifikácia môže byť definovaná ako popis, ktorý dokáže jednoznačne vyjadriť potreby alebo zámer jednej skupiny ľudí skupine ďalšej. Popisy požiadaviek zákazníka, náčrty produktov, zoznamy súčiastok a manuály služieb sú príklady špecifikácií v organizáciách. Špecifikácie pri kustomizovaných produktoch sú najlepšie vyjadrené výberom vopred nadefinovaných modulov (Hvam, Mortensen, Riis, 2008). V tomto prípade sú ako moduly vnímané farby, veľkosti tašiek a dĺžky rukovätí kabeliek. Výber špecifikácií bude prebiehať zákazníkom vo webovej konfiguračnej aplikácii v internetovom obchode, kde budú zobrazované zmeny v reálnom čase.

2.4 Predstavenie zadávateľa

Internetový obchod a konfiguračná aplikácia je vyvíjaná pre Helenu Išovú (Išová, 2016), ktorá sa venuje návrhu a výrobe kabeliek a doplnkov z pravej kože. Zakladá si na kvalitnom remeselnom spracovaní a väčšinu kabeliek vyrába z jedného kusa kože. Momentálne ponúka svoje výrobky na internetovom portáli Fler.cz a taktiež zásobuje butiky v niekoľkých väčších mestách v Českej republike. Vyrába modely aj na zákazku a jej výrobky sú známe vďaka detailnému a veľmi kvalitnému prevedeniu. Produkty zapadajú do vyššej cenovej kategórie a majiteľka chce zákazníkom priniesť ďalšiu pridanú hodnotu, ktorú môžu získať po nákupe jej modelov. Rozhodla sa preto o využitie možnosti kustomizácie svojich produktov. V prvej fáze chce umožniť zákazníkom iba zmenu farby, veľkosti pútka a samotnej kabelky, v budúcnosti však plánuje možnosť ďalších zmien, napríklad voľbu typu materiálu kabelky či špeciálne detaily. Zakladá si však na myšlienke, že každá kabelka musí byť unikátna a musí obsahovať časť jej návrhu, preto plná možnosť zostavenia produktu nie je plánovaná ani do budúcnosti.

2.5 Existujúce riešenia

Ako už bolo spomenuté, firmy môžu možnosťou kustomizácie svojich produktov získať dôležitú konkurenčnú výhodu. Práve toto vedie množstvo firiem ku rozhodnutiu rozšírenia sortimentu ponúknutím kustomizácie. V prípade internetových obchodov sú využívané dynamické webové aplikácie, kde si užívateľ sám zvolí z ponúkaných úprav produktu. Na trhu sa momentálne nachádza niekoľko existujúcich riešení pre produktové konfiguračné aplikácie. Veľké množstvo z nich je ponúkané poskytovateľmi riešení pre internetové obchody, ktorý tieto riešenia ponúkajú ako dodatočný modul kompatibilný s danou platformou. K dispozícii sú však aj platformovo nezávislé riešenia.

Existujúce komerčné riešenia riešia vizualizáciu dvoma spôsobmi. Prvý, menej náročný spôsob využíva na zobrazovanie vizualizácií 2D modely, resp. fotografie

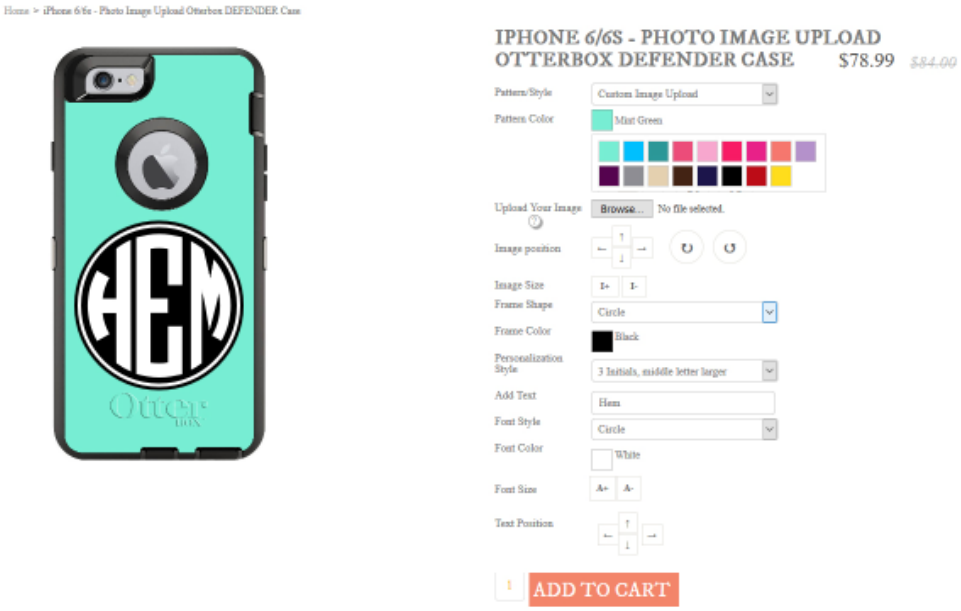
produktov, ktoré sú zložené z modulov. Zaujímavejšia, avšak na zobrazovanie náročnejšia cesta sú 3D modely. Výhodou je realistickejšie zobrazenie a podpora predstavivosti, nevýhodou je však potreba tvorby 3D modelov. Táto možnosť je ideálna pri produktoch, kde sa tvar produktu sám o sebe nemení, ale upravujú sa len farby či pridávajú ďalšie elementy, pričom tvar pôvodného produktu sa nemení. V prípade nášho produktu je však využitie 3D zobrazovania vzhľadom k objemu výroby, počtu modelov a času tvorby 3D modelov neefektívne. Preto som sa rozhodla porovnávať dostupné riešenia, ktoré využívajú 2D modely zobrazovania. Volila som z rôznych cenových kategórií a porovnávala ponúkanú funkcionalitu. Ako prvý som si vybrala Custom Product Designer od firmy Milople. Bližšie som ďalej preštudovala ProductCart Configurator a spomeniem aj riešenie konfiguračnej aplikácie od firmy Fluid.

2.5.1 Milople Custom Product Designer

Toto riešenie je vyvinuté firmou Milople, ktorá navrhuje modely pre najznámejšie e-shopové platformy, ako napríklad Magento alebo Prestashop. K dispozícii je všeobecná galéria produktov, ako napríklad tašky a hrnčeky. V prípade potreby kustomizácie produktu mimo základnej galérie je možné nahrať obrázok produktu a ďalej ho upravovať. Toto riešenie umožňuje zmenu atribútov ako farba produktu, jeho typ, tvar či usporiadanie modulov. Produkt je možné pridať do košíka priamo z aplikácie. Dizajn aplikácie je priemerný, poskytnuté sú iba malé možnosti prispôsobenia rozhrania. Zmeny sa zobrazujú v reálnom čase bez potreby obnovenia celej stránky.

Čo sa týka administrátorskej časti aplikácie, implementovaná je správa objednávok, produktov a dostupných možností úprav. Administrátor si môže kustomizovaný dizajn stiahnuť vo formáte zip, nastaviť možnosti kustomizácie a upravené produkty prezerať v rámci objednávok. Ku objednávkam je zabudovaná možnosť vystavovania faktúr, čo je pri prevádzke e-shopu veľmi užitočné. Poskytnutá je aj možnosť pridania, úpravy a odobrania produktu, úprava a prezeranie stavu objednávok. Mimo aplikácie je k dispozícii zákaznícka podpora a možnosť úpravy riešenia na mieru. Milople konfigurátor je z vybraných existujúcich riešení najmenej finančne náročný.

Pre náš produkt sa riešenie javí ako priemerné. Tento variant je určený predovšetkým na prípravu produktov na tlač, teda pridávanie vlastných obrázkov a textu. K dispozícii je však možnosť zmeny farby a pridávania ďalších modulov, takže možnosti, ktoré je potrebné meniť v našom prípade by boli realizovateľné. Do budúcnosti sa však riešenie javí ako nedostačujúce. Nevýhodou je taktiež závislosť na platforme Magento, čo značne obmedzuje jeho použitie. Výhodou je však kompletná implementácia a zabezpečenie v rámci konfiguračnej aplikácie (milople.com, 2016).



Obrázok 2: Ukážka riešenia od Milople (milople.com, 2016)

2.5.2 ProductCart Configurator

ProductCart ponúka 3 úrovne funkcionality konfiguračnej aplikácie – Standart, Customization a Customization Plus. Pre naše účely je postačujúca verzia Customization, ktorá obsahuje potrebnú funkcionalitu a z porovnávaných riešení predstavuje strednú cestu, čo sa finančnej náročnosti týka. Riešenie poskytuje základnú funkcionalitu ako pridanie ľubovoľného počtu produktov, zobrazovanie reálnych zmien bez potreby obnovenia webu či správu neobmedzeného množstva kategórií. Zaujímavým rozšírením je prezencia nástrojov pre sociálne siete, ktoré umožňujú okamžité zdieľanie upraveného produktu na Facebook či Pinterest. Súčasťou je aj CRM systém obsahujúci všetky informácie o doterajších zákazníkoch, možnosť nastavenia programu odmeňovania a zadávania zľavových kódov. Technickú podporu poskytuje nielen samotný vývojársky tím, ale aj veľká komunita užívateľov zastúpená vo fórach. V sekcii Downloads sú pripravené časti hotového kódu ako prihlasovacie formuláre a vyhľadávacie okná, ktoré si je možné pridať do e-shopu podľa vlastných predstáv. Čo sa samotnej konfigurácie týka, je možné meniť farby produktov, moduly a prakticky čokoľvek, čo pri administrácii aplikácie zadefinujeme. Navyše je pre každú kategóriu možné nastaviť špeciálne konfiguračné stránky. Toto riešenie je pre naše účely zaujímavé, avšak nevýhodou je znova závislosť na Wordpress platforme (productcart.com, 2016).

2.5.3 Customizator od Fluid

Jedno z najznámejších existujúcich riešení poskytuje firma Fluid, ktorá tvorí software pre známe značky ako Reebok alebo Timberland. Konfiguračné aplikácie majú vynikajúce dizajn a sú na vysokej úrovni, čo sa týka užívateľskej prívetivosti. Na zobrazovanie sú využívané aj 3D modely. Riešenie nie je platformovo závislé a podporuje najznámejšie platformy ako hybris a Magento. V našom prípade je toto riešenie vysoko nad rozpočet klientky, avšak môže slúžiť ako inšpirácia a vízia do budúcnosti v rámci ďalšieho vývoja konfiguračnej aplikácie (fluid.com, 2016).

Obrázok 3: Ukážka riešenia od Fluid (fluid.com, 2016)

2.6 Vývoj moderných webových aplikácií

Webové aplikácie sú dynamické webové stránky, ktoré interagujú s užívateľom. Pri vývoji moderných webových aplikácií sa môžeme stretnúť s dvoma hlavnými kategóriami skriptovania a kódovania:

1. Skriptovanie a kódovanie na strane klienta (z angl. client-side) je typ kódu, ktorý je vykonávaný a interpretovaný prehliadačmi. Skriptovanie je bežne viditeľné každým užívateľom webovej stránky.
2. Skriptovanie a kódovanie na strane serveru (z angl. server-side) je typ kódu, ktorý je vykonávaný a interpretovaný na webovom serveri. Toto skriptovanie a kód nie je viditeľný užívateľom ani verejnosťou (Kohan, 2016).

2.6.1 Proces vývoja moderných webových aplikácií

1. Opis webovej aplikácie, jej zámeru a cieľov – jeden z najdôležitejších bodov vývoja aplikácie vôbec, ktorý zahrňuje definovanie jasných cieľov a zámeru aplikácie. Umožňuje správne pochopenie požiadaviek a očakávaní zadávateľa.

2. Stanovenie funkčných a nefunkčných požiadaviek – stanovuje zadávateľ, resp. osoba, ktorá iniciuje a požaduje vývoj aplikácie. Ide o stanovenie presnej funkcionality a technickej špecifikácie, ktoré bude musieť aplikácia spĺňať.
3. Výber technológií, technických špecifikácií, príprava ilustračného diagramu architektúry a štruktúry webovej aplikácie, voľba metodiky a plán ďalšieho vývoja – prebieha výber prostriedkov na dosiahnutie stanovenej funkcionality a plánovanie štruktúry aplikácie.
4. Vizuálna stránka aplikácie, dizajn rozhrania a príprava drôtových modelov – proces začína prípravou vizuálnej šablóny, drôtových modelov alebo jednoduchého náčrtu užívateľského rozhrania a interakcií. V tomto bode je potrebné pripraviť dizajn, ktorý prinesie čo najlepšiu užívateľskú skúsenosť.
5. Štruktúra používanej databázy a vývoj webovej aplikácie – pod týmto bodom je zahrnutý vývoj architektúry aplikácie, dizajn databázovej štruktúry, vývoj aplikačných modulov, a nakoniec finalizácia webovej aplikácie so všetkou funkcionalitou.
6. Testovanie zabezpečenia, výkonu a kompatibility s rôznymi prehliadačmi – prebieha testovanie aplikácie s cieľom definície a adresovania existujúcich chýb.
7. Údržba (Kohan, 2016).

2.7 Používané prostriedky

Pri tvorbe moderných webových aplikácií je k dispozícii celá rada možností a prostriedkov. Najčastejšie je používaný programovací jazyk PHP v kombinácii s databázou. Dynamickú funkcionalitu zabezpečuje použitie Javascriptu, ktorý ponúka celé spektrum možností. Pri implementácii dynamickej webovej aplikácie sa odporúča využívanie frameworkov, ktoré uľahčujú proces písania kódu, nakoľko sami obsahujú existujúce bloky a objekty, ktoré môže vývojár použiť a nemusí začať písať kód od začiatku.

2.7.1 Programové knižnice

Programové knižnice sú kolekcie často používaných funkcií, tried a rutín, ktoré poskytujú uľahčenie vývoja a údržby pre vývojárov poskytnutím zjednodušeného pridania či úpravy funkcionality do aplikácie vo frameworkovej alebo modulárnej štruktúry (Kohan, 2016).

2.7.2 Frameworky pre vývoj webových aplikácií

Frameworky pre vývoj webových aplikácií sú sadou programových knižníc, komponent a nástrojov organizovaných do architektúrneho systému, ktorý umožňuje vývojárom zostavovať a udržovať projekt vývoja webovej aplikácie s použitím rýchleho a efektívneho prístupu. Frameworky sú vyvinuté za účelom urýchlenia a organizovania programovania s používaním organizácie priečinkov, pevne danej štruktúry, dokumentácie, návodov a knižníc s implementovanou funkcionalitou.

Medzi výhody frameworkov patria:

- Programové akcie a logika sú separované od HTML, CSS a súborov obsahujúcich informácie o dizajne. Táto štruktúra umožňuje dizajnérom bez akejkoľvek znalosti kódu vykonávať požadované zmeny v rozhraní či dizajne, a to bez pomoci programátora.
- Stavba je postavená na moduloch, knižniciach a nástrojoch, ktoré ponúkajú možnosť jednoduchého zdieľania a implementácie komplexnej funkcionality a funkcií rýchlym a efektívnym spôsobom.
- Štruktúra podporuje kódovanie s konzistentnou logikou, a zároveň poskytuje možnosť rýchleho zorientovania sa v kóde.
- Frameworky obsahujú veľké množstvo hotovej funkcionality, ktorú je možné prakticky ihneď po inštalácii využívať (Kohan, 2016).

3 Materiály a metodika

Požiadavky zadávateľa na konfiguračnú aplikáciu sú veľmi špecifické, hlavne čo sa týka plánovaných rozšírení do budúcnosti. Z dôvodu chýbajúcej vhodnej, platformovo nezávislej alternatívy konfiguračnej aplikácie na trhu a obmedzeným finančným možnostiam sme sa spolu so zadávateľom rozhodli pre vlastnú implementáciu aplikácie. Tvorba počítačového softwaru je veľmi zložitým procesom prinášajúcim potenciálne riziká. Medzi najčastejšie riziká pri vývoji softvéru patria podľa Addisona a Vallabha (2002) nedostatočné pochopenie požiadaviek, nedostatočný rozpočet alebo zle navrhnutý časový plán či slabá angažovanosť zákazníka, ktorá môže vyústiť až do neakceptovania výsledného produktu, prípadne potrebe jeho kompletného redizajnu. Autori uvádzajú ako možnosti ich elimininácie dôslednú komunikáciu so zákazníkom, detailné dokumentovanie špecifikácií, čo zahŕňa prípravu návrhov a modelov systému. Kvôli tomu je pred samotnou implementáciou potrebné vytvoriť návrh a uistiť sa, že výsledná aplikácia bude spĺňať očakávania zadávateľa a potenciálne nedostatky budú odhalené ešte pred zahájením implementácie.

Po definovaní cieľov a zámeru aplikácie spolu so zadávateľom budú stanovené funkčné a nefunkčné požiadavky na finálny výstup. Ako podpora návrhu bude vytvorený UML diagram, konkrétne diagram prípadov použitia, ktorý slúži na modelovanie funkcionality systému. Pre ukladanie a prácu s dátami využijeme databázový systém MySQL, práca taktiež poskytne návrh štruktúry ukladania dát. Ďalším krokom bude príprava návrhu užívateľského rozhrania, ktoré je dôležité pre zaručenie príjemnej užívateľskej skúsenosti. Pri implementácii internetového obchodu bude použitý PHP framework CodeIgniter. Ten využíva architektúru MVC, ktorá rozdeluje kód na 3 nezávislé časti – modely, kontroléry a pohľady. Použitie Javascriptu umožní viditeľnosť zmien v reálnom čase bez potreby obnovenia stránky. Táto funkcionalita bude podporená využitím Ajaxu a množstva knihovní Javascriptu, ako napríklad jQuery. HTML, CSS a Bootstrap budú použité ako nástroje na interpretáciu výstupov užívateľovi.

3.1 UML

Unified Modeling Language[6] je štandardný modelovací jazyk pre vývoj softvéru. Nakoľko sa jedná o formálny jazyk, každý jeho element má pevne definovaný špecifický význam a interpretáciu. Je používaný za účelom tvorby modelov reality, čím je umožnené lepšie pochopenie riešeného problému. Hlavným nástrojom UML je diagram, ktorý je možné vnímať ako okno do modelu. Podľa zvoleného typu diagramu je poskytnutý konkrétny pohľad namiesto zobrazovania všetkých informácií, ktoré by mohli viesť k nesprávnej interpretácii skutočností (Pilone, Pitman, 2005). UML 2.0 definuje 13 typov diagramov, ktoré sú rozdelené do dvoch hlavných kategórií a jednej podkategórie:

[6] Ďalej len UML.

- Diagramy štruktúr zahŕňajú 6 diagramov reprezentujúcich statické aspekty systému. Na ich znázornenie sú využívané triedy, rozhrania, objekty, komponenty a uzly. Tieto diagramy sa používajú hlavne v dokumentácii softvérovej architektúry. Napríklad, najčastejšie používaný diagram tried prezentuje reálne objekty vo všeobecnej forme.
- Diagramy správania zachytávajú dynamické aspekty systému. Ich využitie je možné nájsť pri popisovaní funkcionality softvérových systémov. Diagramy interakcie sú podkategóriou diagramov správania, ktoré bližšie špecifikujú interakciu medzi jednotlivými prvkami (uml.org, 2005).

UML môže byť používaný v rôznych prípadoch, najčastejšie však pri návrhu softvéru, zaznamenávaní systémových detailov a dokumentácii existujúceho systému, procesu či organizácie (Pilone, Pitman, 2005).

3.2 PHP

Hypertext Preprocessor[7] je skriptovací programovací jazyk používaný pri vývoji dynamických webových stránok. Skripty sú vykonávané na strane serveru, kde je generované HTML a ku klientovi sa dostáva len výsledok operácie (Macintyre, 2010). PHP môže byť použité vo všetkých hlavných operačných systémoch a taktiež podporuje väčšinu webových serverov. Ďalšou dôležitou výhodou je široká podpora databázových systémov, ako napríklad MySQL. Za momentálne stabilné verzie sa považujú verzie PHP 7.0.13 a PHP 5.6.28 (php.net, 2016).

3.3 MySQL

MySQL je open-source relačný databázový systém, ktorý umožňuje technológiam PHP a Apache sprístupňovať a zobrazovať dáta vo formáte podporovanom internetovými prehliadačmi (Naramore, 2006). Na zadávanie príkazov sa používa štruktúrovaný dotazovací jazyk SQL. Pri začiatočnom vývoji MySQL bol kladený dôraz na rýchlosť, ktorá bola dosahovaná aj za cenu absencie dôležitých funkcií. Avšak nedostatky predošlých verzií, a to pridanie cudzích kľúčov, pohľadov a ostatných prispôsobiteľných indexných typov, boli vyriešené a odstránené v najnovších verziách, čo zvýšilo používanosť a obľúbenosť systému medzi programátormi (mysql.com, 2016).

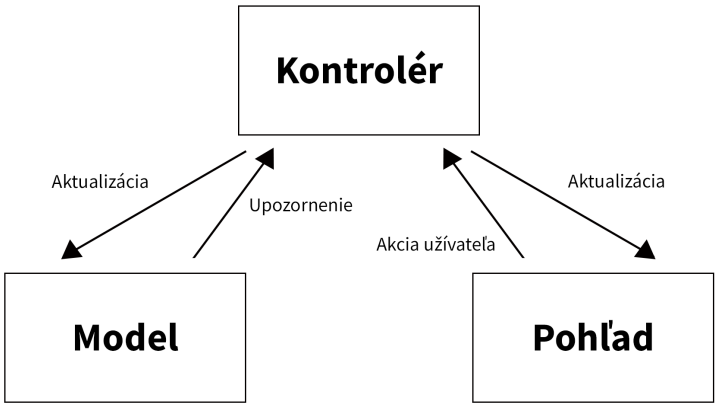
3.4 MVC architektúra

Definíciu MVC architektúry prvýkrát naformuloval profesor Trygye Reenskaug v roku 1979. MVC, alebo Model-View-Controller je softwarová architektúra, ktorá rozdeluje zdrojový kód do troch na sebe nezávislých vrstiev a napomáha tak kontrole veľkých a komplexných projektov. Zmena v jednej vrstve má potom na ostatné vrstvy minimálny dopad, čo viditeľne uľahčuje spoluprácu viacerých vývojárov na

[7] Ďalej len PHP.

jednom projekte. Základný význam MVC je poskytnutie spojenia medzi užívateľským mentálnym modelom a digitálnym systémom, ktorý existuje v počítači. Ideálne riešenie podporuje užívateľský dojem priameho prístupu a manipulácie s doménovými informáciami. Štruktúra je užitočná v prípade, že užívateľ používa rovnaké elementy modelu pravidelne v rôznych obsahoch a v rôznych situáciach (Reenskaug, 1979). Každá vrstva má svoj osobitný význam:

- Model reprezentuje štruktúru dát. Jeho triedy obsahujú funkcie, ktoré pomáhajú upravovať, vkladať a získavať informácie z databázy. Získané informácie sú následne odoslané do kontroléru.
- Pohľad[8] predstavuje informáciu, ktorá je prezentovaná užívateľovi. Najčastejšie je prezentovaný v podobe webovej stránky tvorenej generovaným HTML kódom.
- Kontrolér[9] slúži ako spojenie, resp. komunikačný most medzi modelom a pohľadom, poprípade inými prostriedkami, ktoré sú potrebné k vykonávaniu HTTP požiadavku a generovaniu webovej stránky. Spracúva potrebné dáta z modelu a neskôr ich predáva pohľadu na zobrazenie užívateľovi (codeigniter.com, 2016).



Obrázok 4: Model MVC architektúry (chrome.com, 2016)

3.5 CodeIgniter

Codeigniter je open-source framework určený na vývoj dynamických webových stránok s použitím jazyka PHP. Jedným z cieľov vývoja frameworku je umožnenie rýchlejšieho vývoja projektov. Dosiahnutie cieľa je zabezpečené množstvom knižníc obsahujúcich riešenia klasických problémov a úloh a taktiež jednoduchým rozhraním

[8]Z angl. view.
[9]Z angl. controller.

a logickou štruktúrou, ktoré uľahčujú prístup ku informáciam z knižníc. Ďalšou výhodou je rozsiahla dokumentácia a aktívna komunita užívateľov, ktorí spolu komunikujú vo fórach. CodeIgniter na svojich stránkach prezentuje informáciu o nenáročnosti frameworku. Na jeho inštaláciu je oproti iným konkurenčným frameworkom potrebné iba malé množstvo knižníc. Dodatočné knižnice sú načítané až po zadaní požiadavku, čo zaručuje rýchlosť systému (Upton, 1995).

Je založený na populárnej MVC architektúre, ktorá rozdeľuje kód na tri nezávislé časti – modely, kontroléry a pohľady. Framework má však ku tejto architektúre veľmi voľný prístup. Pokiaľ pri vývoji aplikácie nie je potrebná separácia kódu alebo by spôsobovala prílišnú komplikáciu, užívateľ môže modely ignorovať a pri vývoji použiť kontroléri a pohľady v minimálnej miere (codeigniter.com, 2016). CodeIgniter taktiež umožňuje zabudovanie vlastných existujúcich skriptov alebo budovanie jadrových knižníc pre systém.

CodeIgniter bol vytváraný z technického a architektúrneho uhla pohľadu s týmito víziami:

- Dynamická konkretizácia (z angl. dynamic instantiation) pomenúva postup, keď komponenty sú načítané a programy vykonávané až po prijatí požiadavku. Vykonávané sú potom iba nevyhnutné procesy a zdroje, čo zaručuje plynulosť systému už od počiatočných fáz. Udalosti vyvolané HTTP požiadavkom ovplyvňujú konečný výsledok.
- Minimálne väzby (z angl. loose coupling) predstavujú stupeň v akom sú systémové komponenty na sebe závislé. Čím menej sú komponenty na sebe závislé, tým flexibilnejší systém je.
- Jedinečnosť komponent (z angl. component singularity) definuje level špecializácie komponent. Vo frameworku CodeIgniter je každá trieda a jej funkcia čo najpodrobnejšie definovaná a špecializovaná za účelom zvýšenia jej užitočnosti a prevencii proti duplicite (codeigniter.com, 2016).

CodeIgniter je dynamicky konkretizovaný, s minimálnymi väzbami komponent, ktoré majú vysokú jedinečnosť, resp. špecializáciu. Hlavným cieľom je zaručiť jednoduchosť, flexibilitu a vysoký výkon s čo najmenšou stopou. Taktiež podporuje takmer všetky zdieľané hostingové platformy. Pri vývoji webovej aplikácie je potrebné použitie databázy, framework podporuje najznámejšie a najpoužívanejšie databázové platformy, medzi inými aj MySQL (codeigniter.com, 2016).

Framework bol predstavený v roku 2006. Aktuálna verzia CodeIgniter 3.x je aj naďalej aktívne vyvíjaná a ďalej rozširovaná. Jeho zdrojový kód spravuje GitHub, a podobne ako pri predošlej verzii 3.0rc, framework je certifikovaný pod MIT licenciou (codeigniter.com, 2016).

3.6 Javascript

Javascript je interpretovaný programovací jazyk podporujúci možnosti používania objektov. Jeho syntax je podobná jazyku C, C++ a Java. Najčastejšie sa používa

vo webových prehliadačoch, nakoľko jadro jazyka je rozšírené o možnosť používania objektov, ktoré umožňujú interakciu skriptov s užívateľom, kontrolu webového prehliadača a úpravu obsahu dokumentu, ktorý sa zobrazuje v okne prehliadača. Táto stabilná verzia Javascriptu spúšťa skripty v rámci HTML stránok. Javascript je vnímaný ako client-side, pretože skripty prebiehajú na strane užívateľa a nie na strane webového serveru (Flanagan, 2006).

JavaScript umožňuje tvoriť vysoko responzívne prostredie, ktoré zlepšuje užívateľskú skúsenosť a poskytuje dynamickú funkcionalitu bez potreby čakania na reakciu zo strany serveru. Medzi moderné využitia tohoto programovacieho jazyka môžeme zaradiť kontrolu dostupnosti užívateľského mena pri registrácii, automatické plnenie formulárov, viditeľnosť zmien v reálnom čase či zvýrazňovanie špecifických sekcií webových stránok (Flanagan, 2006).

3.7 Bootstrap, HTML5 a CSS3

Bootstrap je open-source front-end webový framework určený na vývoj webových stránok a webových aplikácií, ktorý je vyvíjaný spoločnosťou GitHub. Vývoj webu zjednodušuje široké spektrum rôznych HTML a CSS šablón, napríklad galéria ikon, navigačné komponenty, blogy či grafické šablóny webu. Zaujímavé rozšírenia ponúkajú aj JavaScript komponenty, napríklad tzv. popovers. Bootstrap bol originálne vyvíjaný firmou Twitter ako projekt Twitter Blueprint pre interné účely, neskôr bol však pre veľkú popularitu medzi programátormi sprístupnený aj širokej verejnosti. Najaktuálnejšia verzia momentálne je Bootstrap v4.0.0-alpha.5 (github.com, 2016).

3.8 Ďalšie použité nástroje

Toastr
Je Javascript knihovňa pre neblokované notifikácie. K jej využitiu je potrebné jQuery. Hlavným cieľom je vytvorenie jednoduchej jadrovej knihovne, ktorá môže byť kustomizovaná a rozširovaná (getbootstrap.com, 2016).

Ajax
Ide o skupinu technológií na vývoj webu, ktoré sú používané na strane klienta a umožňujú vývoj asynchrónnych webových aplikácií. Skladá sa z XML, ktorý získava dáta z webového serveru a Javascriptu, ktorý získané dáta zobrazuje. Ajax umožňuje webovým stránkam asynchrónne obnovenie vďaka tomu, že dáta si s webovým serverom vymieňa „za oponou“. To znamená, že je možné aktualizovať časti stránky bez potreby obnovenia celej stránky (w3schools.com, 2016).

Jquery
Jquery je rýchla, malá a na rozšírenia bohatá JavaScript knihovňa. Vďaka jednoduchému API zjednodušuje akcie ako manipuláciu s HTML dokumentom, správu

udalostí, animáciu a Ajax. Príjemnou výhodou je podpora širokého spektra prehliadačov, a taktiež mnohostrannosť a flexibilita knihovne (jquery.com, 2016).

Apache
Je open-source http server pre moderné operačné systémy, a to UNIX a Windows (apache.org, 2016). Apache je vyvíjaný a udržovaný otvorenou komunitou vývojárov spadajúcich pod Apache Software Foundation. Tento server podporuje množstvo funkcií, z ktorých je mnoho implementovaných ako kompilované moduly, ktoré rozširujú pôvodnú funkcionalitu.

4 Výsledky

4.1 Definícia funkčných a nefunkčných požiadaviek

Správna a podrobná definícia požiadaviek na aplikáciu minimalizuje zmeny, ktoré je potrebné vykonávať v systéme po zahájení vývoja (Howard, 2008). Požiadavky je možné rozdeliť na dve hlavné kategórie – funkčné a nefunkčné požiadavky. Funkčné požiadavky popisujú funkcionalitu systému, zatiaľ čo nefunkčné požiadavky stanovujú obmedzenia a vlastnosti systému. Zber požiadaviek by mal zásadne prebiehať so zadávateľom vývoja aplikácie a mala by mu byť venovaná patričná pozornosť, nakoľko zásadne ovplyvňuje celý proces vývoja aplikácie.

4.1.1 Funkčné požiadavky

Pre lepšiu orientáciu sú funkčné požiadavky rozdelené na požiadavky popisujúce funkcionalitu e-shopu a požiadavky, ktoré popisujú funkcionalitu konfiguračnej aplikácie. V rámci každej kategórie sa nachádzajú podkategórie bližšie definujúce skupinu požiadaviek, napríklad práva užívateľov.

Požadovaná funkcionalita e-shopu:

- Prehliadanie produktov – Zobrazený bude obrázok produktu a tlačítko Detail. Po kliknutí na tlačítko sa fotka produktu zväčší, zobrazí sa jeho názov, cena, popis a tlačítka Objednať a Konfigurovať. Po kliknutí na tlačítko Objednať sa produkt pridá do košíka. Prechod do konfiguračnej aplikácie sa iniciuje po kliknutí na tlačítko konfigurovať. V rámci prehliadania produktov je možné aj stránkovanie.
- Prihlásenie a registrácia užívateľa – Užívateľ sa môže registrovať. Pri registrácii zadáva meno, priezvisko, e-mail a heslo. Prihlasovanie prebieha pomocou e-mailu a hesla. Po prihlásení má k dispozícii možnosť prehliadať svoje objednávky a vkladať produkty do košíka.
- Nákupný košík – Prihlásenému užívateľovi sa zobrazujú produkty v košíku. Následne má možnosť meniť množstvo produktov alebo produkt jednoducho z košíka vymazať. Košík na základe obsahu ráta konečnú sumu objednávky.
- Objednávka – Produkty z košíka sa dajú priamo objednať. Suma poštovného je fixné daná a platba je možná iba prevodom.

Rozšírená funkcionalita e-shopu pre admina:

- Správa produktov – Admin má možnosť produkty mazať a upravovať ich atribúty. Konkrétne obrázok produktu, jeho názov, popis a cenu. Taktiež má možnosť do systému pridať nový produkt.
- Správa užívateľov – Správa užívateľov zahrňuje kompetenciu na odstránenie užívateľského účtu či úpravu jeho atribútov.

- Správa objednávok – Admin si môže zobraziť výpis objednávok v systéme s obrázkom produktov, konečnou cenou a stavom objednávky. Stav objednávky môže byť zaplatená/ nezaplatená.

Požadovaná funkcionalita aplikácie:

- Zobrazovanie zmien produktov – Aplikácia zobrazuje zmeny v reálnom čase bez potreby obnovenia stránky. Zmeny sa prejavia ihneď po kliknutí na zvolenú variantu.
- Výber požadovaných atribútov – Zákazník má možnosť zmeny farby kabelky u špecifických modelov (záleží od modelu). Po kliknutí na zvolenú farbu sa zobrazí kabelka v danej farbe. Zákazník má možnosť nastavenia veľkosti kabelky a zmeny veľkosti pútka. Zmeny sa však nezobrazia, nakoľko sú príliš špecifické a závislé na konkrétnom modely.
- Rátanie ceny – Po každej zmene sa zmení aj cena. Každá varianta má fixnú cenu, ktorá sa pripočítava k základnej cene kabelky pred kustomizáciou. Ošetrené budú implicitné veľkosti kabeliek a pútok, pri zvolení tejto varianty sa cena nemení.
- Možnosť pridania do košíka – Po finalizácii konfigurácie produktu zákazník môže zvoliť možnosť objednania a výsledný produkt bude pridaný do košíka.

4.1.2 Nefunkčné požiadavky

Dôležitou nefunkčnou požiadavkou je jednoduchosť používania konfiguračnej aplikácie a možnosť viditeľnosti zmien v reálnom čase. Systém taktiež musí umožňovať prístup do aplikácie bez webové rozhranie. E-shop nebude podporovať rôzne jazykové mutácie a bude používaná iba jedna mena – CZK.

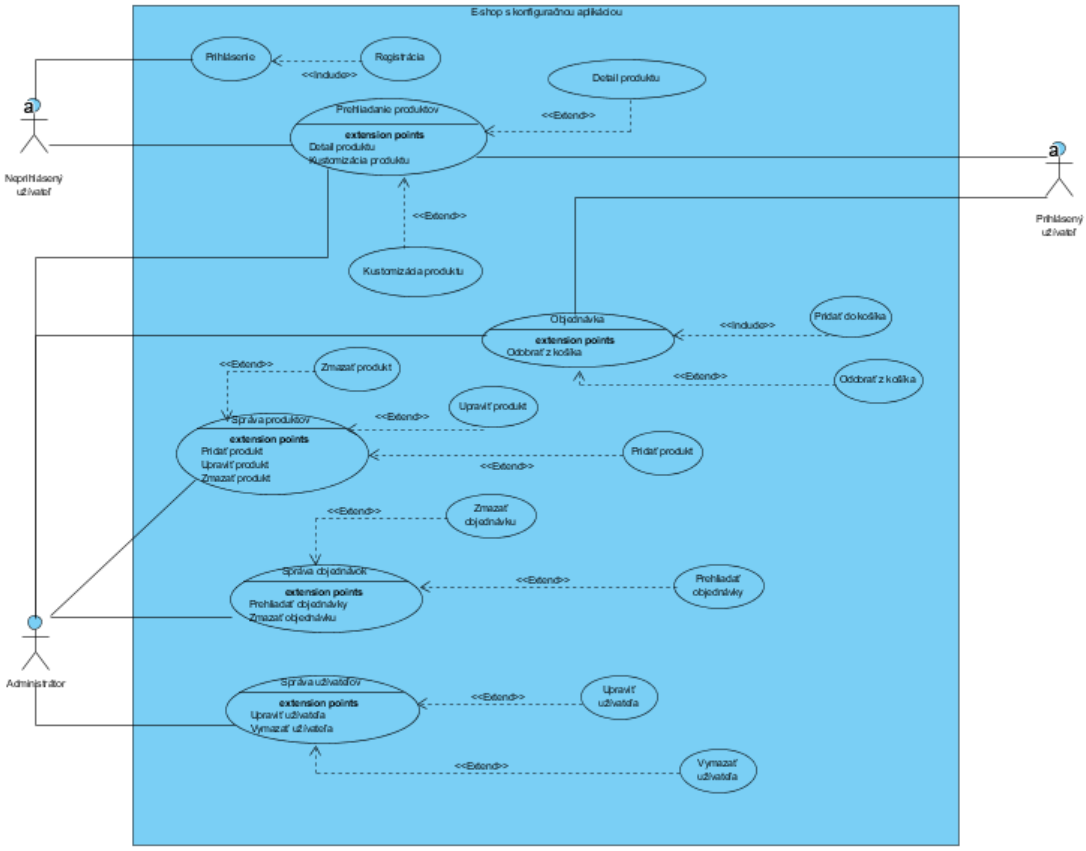
4.2 Diagram prípadov použitia

Po definovaní funkčných a nefunkčných požiadaviek na internetový obchod a aplikáciu bol zostavený diagram prípadov použitia. Jedná sa o diagram z kategórie UML diagramov správania, ktoré vizualizujú, špecifikujú a dokumentujú dynamické aspekty systému. Diagram prípadov použitia poskytuje možnosť modelovania systémovej funkcionality a aktérov, ktorý s touto funkcionalitou pracujú (visual-paradigm.com, 2016).

Diagram sa skladá z konkrétnych prípadov použitia, aktérov a asociácií medzi nimi. Prípad použitia môžeme definovať ako špecifikáciu skupiny akcií vykonávaných systémom vedúcich k pozorovateľným výsledkom, ktoré sú dôležité pre jedného alebo viacerých aktérov alebo účastníkov systému. Vzťah medzi aktérom a prípadom použitia nazývame asociácia. Tá korešponduje so sekvenciou akcií, ktoré prebiehajú medzi aktérom a prípadom použitia za účelom vykonania danej funkcionality. Aktéri sú entity, ktoré interagujú so systémom. Aj napriek faktu, že aktérmi sú najčastejšie osoby, aktérom sa môže stať prakticky čokoľvek, čo potrebuje vymieňať informácie

so systémom. Aktérom môže byť človek, počítačový hardware alebo iné systémy (visual-paradigm.com, 2016).

Všetky spomenuté notácie sú v tomto prípade uložené v systéme Internetový obchod. Každý užívateľ, ktorý príde do styku so systémom, je definovaný aktérom. Aktérom je teda neprihlásený užívateľ, prihlásený užívateľ, ktorý má o niečo väčšie právomoci a nakoniec admin, ktorý môže spravovať produkty, objednávky či užívateľov. Podrobnú funkcionalitu pre každého aktéra môžeme vidieť na obrázku nižšie.



Obrázok 5: Diagram prípadov použitia

Neprihlásený užívateľ

Neprihlásený užívateľ je každý návštevník stránky, ktorý nemá užívateľský účet, teda nevykonal registráciu alebo sa neprihlásil pomocou svojich prihlasovacích údajov, ktoré sa skladajú z e-mailu a hesla. Užívateľ si produkty môže prehliadať, zobrazovať ich detail a konfigurovať, nemôže si ich však vložiť do košíka, táto možnosť sa mu ani nezobrazuje. Funkcionalita objednávania je pre tohoto aktéra nedostupná.

Prihlásený užívateľ
Predstavuje aktéra, ktorý sa prihlásil pomocou svojich prihlasovacích údajov, teda už vykonal registráciu. Tento aktér si môže produkty prehliadať, zobrazovať ich detail a kustomizovať ich. Produkt je po finalizácii kustomizácie možné vložiť do košíka a objednať. Užívateľ si zároveň môže prehliadať produkty v košíku, mazať ich a meniť ich množstvo. Taktiež je k dispozícii možnosť prehliadania jednotlivých objednávok.

Admin
Admin má k dispozícii všetku funkcionalitu ako prihlásený užívateľ a zároveň má prístup do správy produktov, objednávok a užívateľov. Pri správe produktov môže pridávať nové produkty, meniť atribúty existujúcich produktov a taktiež môže produkty mazať. Správa objednávok zastrešuje zoznam objednávok, ich prehliadanie a mazanie. Detail objednávky obsahuje informáciu o uhradení, resp. neuhradení objednávky. Správa užívateľov zase poskytuje možnosť mazania užívateľov, úpravy ich mena, priezviska, e-mailu a role, teda nastavovanie administrátorov systému.

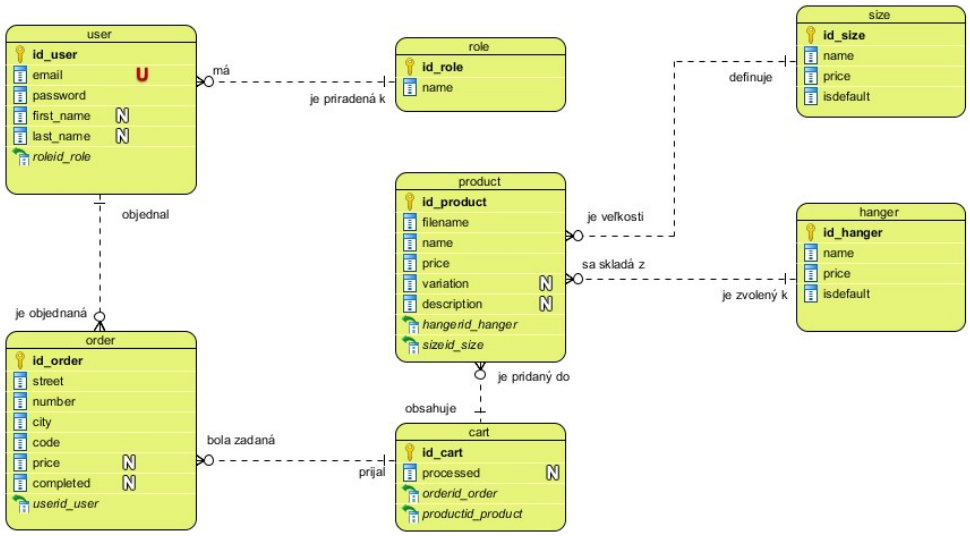
4.3 Entitno-relačný diagram databázy

Entitno-relačný diagram databázy[10] je reprezentácia dát v rámci domény. Pozostáva z entít a reláciami medzi jednotlivými entitami. Entita môže predstavovať hmotný, fyzický objekt ako napríklad produkt alebo objednávka. Entita extrahuje objekty významné pre doménu problému a pre vývoj systému. Každá entita sa skladá zo skupiny stĺpcov, kde platí, že každý stĺpec reprezentuje jednu vlastnosť, resp. informáciu o entite. Každá entita musí mať aspoň jeden unikátny atribút, na základe ktorého môže byť presne identifikovaná. Tento atribút sa nazýva primárny kľúč, najčastejšie sa jedná o id entity. Entity sú spojené reláciami a ich najčastejšími typmi sú 1:1, 1:M a M:N. Skutočný význam používania relácií je vyjadrenie spôsobu, akým sú entity spojené (visual-paradigm.com, 2016). Napríklad užívateľ môže mať žiadnu alebo viacero objednávok, avšak objednávka musí mať práve jedného priradeného užívateľa.

Na obrázku ER diagramu môžeme vidieť entitu user popisujúcu zákazníka. Ten má priradený primárny kľúč `id_user` a ukladáme jeho unikátny e-mail, heslo, meno a priezvisko. Každý užívateľ má priradenú jednu rolu a to user alebo admin. Každá rola môže mať jedného alebo viacero priradených užívateľov, čo znamená, že administrátor nemusí byť iba jeden. Entity size a hanger definujú konkrétne možnosti úprav v rámci kustomizácie a sú bližšie popísané atribútmi name, price a isdefault. Cena za každú variantu je fixná. Tieto bližšie definujú produkt, ktorý má práve jeden druh veľkosti a pútka. Ten obsahuje v atribúte variation názov súboru upravenej fotografie v prípade, že produkt bol kustomizovaný. V opačnom prípade je definovaná hodnota NULL. Produkty môžeme pridávať do košíka. Do entity order

[10]Taktiež ERD či ER diagram.

ukladáme informácie o konkrétnej objednávke, resp. o adrese a zaplatení objednávky. Každá objednávka môže byť objednaná viacerými zákazníkmi a je pridávaná do košíka. Ten môže obsahovať viacej objednávok.



Obrázok 6: Entitno-relačný diagram

4.4 Návrh grafického užívateľského rozhrania

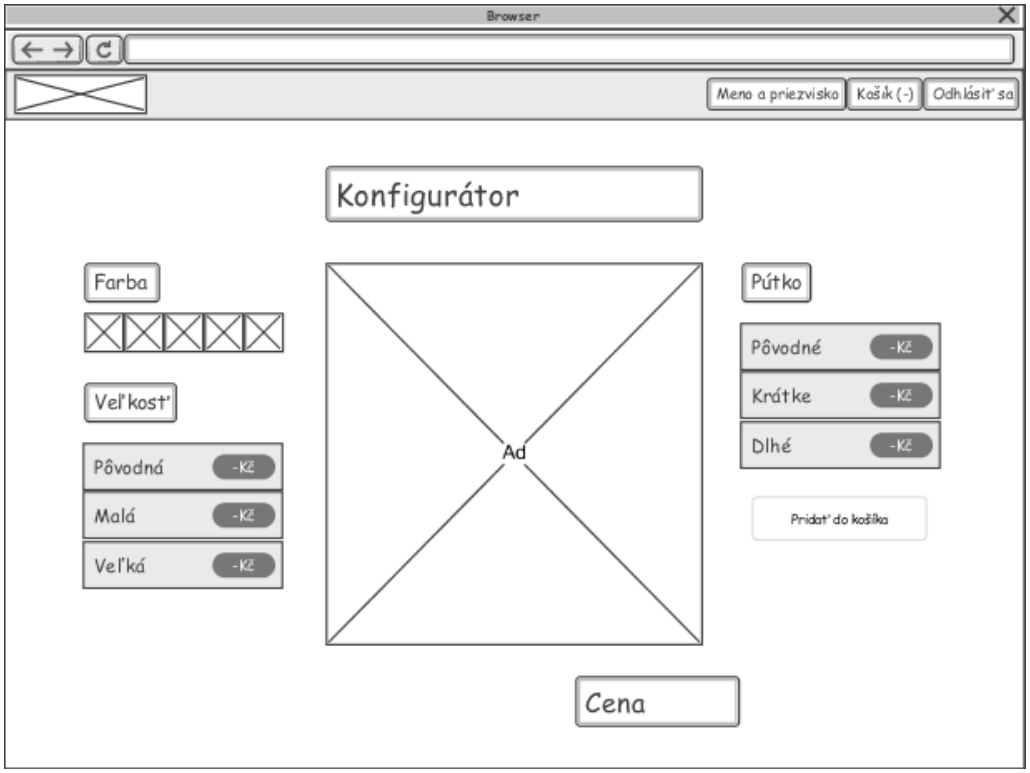
Grafické užívateľské rozhranie[11] je časť počítača a jeho softvéru, ktorú môže užívateľ vidieť, počuť, cítiť či inak rozumieť a vnímať. Najlepšie rozhranie je to, ktoré neprekáža a umožňuje sa užívateľovi sústrediť na informáciu a úlohu namiesto sústredenia sa na mechanizmus interpretácie tejto informácie či úlohy. Užívateľom je rozhranie často vnímané ako samotný systém, pretože patrí medzi jednu z mála viditeľných častí systému. Akcie v rámci GUI sú zvyčajne prevádzané priamou manipuláciou s grafickými elementami (Galitz, 2007).

Ako už bolo spomenuté, návrh grafického užívateľského rozhrania je jeden z krokov pri vývoji webových aplikácií. Návrh zobrazujeme najčastejšie pomocou drôtových modelov, ktoré boli vytvorené v softvéri . Na tento krok som sa rozhodla využiť softvér Balsamiq. Pri príprave užívateľského rozhrania bol kladený dôraz na jednoduchosť, vzdušnosť a intuitívne prostredie. Dôraz bol kladený hlavne na GUI konfiguračnej aplikácie, ako bolo stanovené v nefunkčných požiadavkách.

[11] Taktiež GUI.

4.4.1 Návrh rozhrania konfiguračnej aplikácie

Do konfiguračnej aplikácie sa užívateľ dostane preklikom z detailu produktu. Prihlásenie nie je podmienkou vstupu do aplikácie. Najdôležitejšia a primárna časť aplikácie je obrázok produktu, ktorý sa po výbere z možností mení. Po bokoch sú umiestnené možnosti v troch kategóriách, a to farba, veľkosť a pútko. Na ľavej strane je hneď ponúknutá možnosť výberu farby, pretože to je často rozhodujúci faktor podľa skúseností zadávateľa. Ako ďalšia je ponúknutá možnosť výberu veľkosti tašky. Vždy je uvedený rozmer a príplatok, ktorý bude účtovaný za danú variantu. Napravo sa dostávame k možnosti výberu pútka. Znova je pri každej variante uvedená aj cena. Po finalizácii kustomizácie vedie prostredie zákazníka k tlačidlu Pridať do košíka, ktoré sa nachádza za poslednou možnosťou výberu pútka. Pod produktom je zobrazená cena, ktorá sa automaticky prerátava.

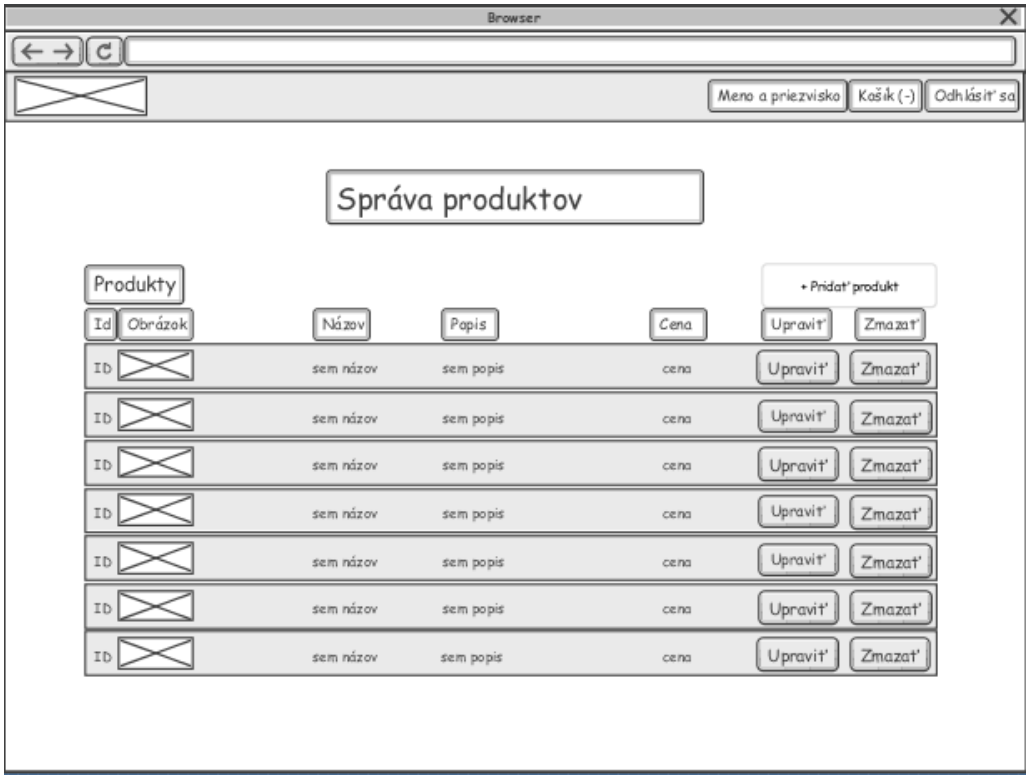


Obrázok 7: Drôtový model rozhrania konfiguračnej aplikácie

4.4.2 Návrh rozhrania správy produktov

Do správy produktov má prístup iba administrátor systému, musí byť teda predtým prevedené prihlásenie užívateľa s rolou Administrátor. Nad všetkými produktami je umiestnené tlačítko Pridať produkt. Vo vyskakovacom okne sa vypĺňuje najskôr názov produktu, popis, cena a nakoniec sa vyberá obrázok. Platí, že každý produkt je zastúpený jedným riadkom, kde je úplne napravo uvedené id produktu, fotografia,

ďalej názov produktu, jeho popis, cena a tlačítka Upraviť a Zmazať. Prechádzaním kurzorom nad obrázkom je tento zväčšený a riadok sa rozšíri. Po kliknutí na tlačítko Upraviť produkt sa zobrazí vyskakovacie okno s formulárom, kde je možné upravovať atribúty názov, popis a cena produktu. Pod kolonkou s cenou je tlačítko Uložiť zmeny. V pravom dolnom rohu je tlačítko Zavrieť. Po zvolení možnosti zmazania produktu sa zobrazí okno s overením zmazania.



Obrázok 8: Drôtový model rozhrania správy produktov

4.5 Štruktúra aplikácie

Každá aplikácia sa skladá zo základných stavebných blokov. V rámci frameworku CodeIgniter, ktorý delí kód na tri nezávislé časti sa aplikácia skladá z modelov, pohľadov a kontrolérov.

4.5.1 Modely

Aplikácia obsahuje sedem modelov, ktoré obsahujú funkcie, ktoré získavajú a následne upravujú informácie z databázy. Implementácia košíka je ošetrená v modeli Cart. Ten obsahuje funkciu na vkladanie produktu do košíka, ktorý ukladá id produktu, jeho variantu, typ pútka, veľkosť a užívateľa, ktorý produkt plánuje zakúpiť. Implementované je taktiež potvrdenie objednávky, prehliadanie obsa-

hu a mazanie produktov z košíka. Model Order obsahuje funkcie na pridávanie, prehliadanie a mazanie objednávok. Zaujímavejšiu funkcionalitu obsahuje model Product, kde je okrem pridávania, prehliadania, upravovania a mazania produktov ošetrené aj stránkovanie zobrazovania. Táto implementácia je v rámci funkcií `get_number_of_pages` a `get_products_with_limit`, kde je nastavený počet produktov na jednu stranu. Prihlasovanie, odhlasovanie, správa užívateľov a získavanie aktuálneho užívateľa a overovania role administrátora je riešené v rámci modelu User. Modely Color, Hanger a Size obsahujú funkcie, ktoré zabezpečujú získavanie všetkých dostupných variánt z databázy pomocou jednoduchých SQL dopytov.

4.5.2 Kontroléry

Kontroléry spracúvajú potrebné dáta z modelu a predávajú ich pohľadom, ktoré ich zobrazujú užívateľovi. Nájdeme ich v priečinku controllers. Všetky sú potomkami `MY_Controller`, ktorý sme vytvorili kvôli šablónovaciemu systému. Ten overuje stav, či je užívateľ prihlásený a podľa toho posúva informácie do pohľadov k zobrazovaniu. Kontrolér sa stará aj o predanie informácií pri odhlásení užívateľa. Väčšina kontrolérov obsahuje funkciu `index`, ktorá predáva dáta do funkcie `layout` a aktivuje požadovaný pohľad. Kontrolér Home okrem toho posúva informácie o stránkovaní. Najviac funkcionality obsahuje kontrolér Ajax, ktorý pripravuje dáta pre pohľady vo formáte json, slúžiace ku generovaniu stránky bez potreby obnovenia.

4.5.3 Pohľady

Pohľady majú na starosti zobrazovanie výstupov. V rámci pohľadov bol naimplementovaný aj vlastný šablónovací systém. Tento bude podrobne popísaný neskôr. Pohľady obsahujú javascripty, ktoré nahrávajú potrebné dáta a HTML časť pohľadov sa stará o formátovanie výstupov. Pohľad Customizator obsahuje implementáciu samotnej konfiguračnej aplikácie. Taktiež obsahuje funkciu `calculatePrice`, ktorá sa stará o zobrazovanie a rátanie ceny pri výbere jednotlivých úprav.

4.6 Implementácia ukladania a zmeny farieb v konfiguračnej aplikácii

Ukladanie a zmena farieb v rámci konfiguračnej aplikácii prebieha na základe názvu súborov, pod ktorými sú uložené jednotlivé farebné variácie. Každý produkt má uložený atribút `filename`, v ktorom sa nachádza názov súboru s príponou, napríklad `1482083109.jpg`. Všetky farebné variácie sú uložené v priečinku Color variations pod názvami, ktoré sa skladajú s filename produktu, ktorého sa variácia týka, podčiarkovníka a hexadecimálnej hodnoty farby variácie, napríklad `1482083109_772d16.jpg`. V rámci konfiguračnej aplikácie je možné meniť farebné variácie konkrétneho produktu len v prípade, že v priečinku existujú súbory s filename zvoleného produktu.

Samotné získavania a zobrazovanie farebných variácií sa nachádza v pohľade Customizator. Ten sa stará o nahrávanie dostupných veľkostí kabeliek, pútok a rátanie ceny. Taktiež zabezpečuje zobrazovanie všetkých farebných variánt, z ktorých si následne užívateľ pri procese kustomizácie vyberá. Tie získavame pomocou Ajaxu v metóde pracujúcej s filename produktu. Volá funkciu, ktorá postupne prechádza každú farebnú variáciu v rámci poľa `colorVariations`. Do premennej `color` sa následne ukladá reťazec so znakom „#“ a časťou názvu súboru farebnej variácie bez časti pred podčiarkovníkom, teda z pôvodného názvu `1482083109_772d16.jpg` sa stane po tomto kroku `#772d16.jpg`. V ďalšom kroku delíme novozískaný názov podľa bodky a ukladáme iba prvú časť. Získame tým hexadecimálnu hodnotu farby. Túto použijeme v ďalšom kroku pri generovaní HTML kódu ako farbu pozadia pre prvky zoznamu farieb. Nakoniec je ako aktívna farba zvolená implicitná farba, ktorá bola zobrazená v galérii pred zvolením možnosti konfigurácie.

```
$.ajax({
    url: '',
    type: "POST",
    data: {pattern: product.filename.split('.')[0]},
    dataType: 'json',
    success: function (response) {
        colorVariations = response;
        for (var i~= 0; i~< colorVariations.length; i++) {
            // splitting out filename
            var color = "#" + colorVariations[i].split('_')
               [1];
            // splitting out suffix
            color = color.split('.')[0];
            $("#colors ul").append("");
        }
        $(".defaultColor").addClass("activeColor");
    },
    error: function (xhr, ajaxOptions, thrownError) {
        alert(xhr.status);
        console.log(xhr.responseText);
        alert(thrownError);
    }
});
```

Neskôr v kontroléri nastavujeme reakciu na voľbu farby užívateľom. V prípade, že farba je rovnaká, ako implicitná farba, obrázok sa nezmení. Pri zvolení inej farby sa obrázok vyberá zo zložky colorVariations a obsahu premennej `variationfilename`. Zvolená varianta sa ukladá vo forme názvu súboru do atribútu `variation`, ktorý je evidovaný v rámci entity product.

4.7 Získavanie dát zo serveru pomocou Ajaxu

Jquery knihovňa, ktorá je v práci používaná obsahuje veľké množstvo Ajax možností. V práci je Ajax použitý za účelom získavania dát zo serveru bez potreby znovunačítania celej stránky. V tomto prípade sa aktualizuje iba časť stránky bez toho, aby to narušilo užívateľskú skúsenosť. Pre plynulý chod a využitie Ajaxu je najskôr potrebné pripraviť dáta vo formáte json. O to sa stará kontrolér Ajax obsahujúci funkcie, ktoré dáta najskôr načítajú, uložia do polí a nakoniec vrátia výstup. Ten je upravený funkciou `json_encode`, ktorá vracia json reprezentáciu dát, v prípade, že prebehla úspešne, inak vracia hodnotu FALSE. Kontrolér predáva dáta do pohľadov. Tie obsahujú javascripty, ktoré využívajú Ajax na komunikáciu so serverom. Na obrázku nižšie môžeme vidieť javascript obsahujúci použitie Ajaxu za účelom vykonania http požiadavku. Ten obsahuje niekoľko atribútov, bližšie definujúcich požiadavok. Atribút `url` používa php príkaz na získanie aktuálnej url adresy a zavolanie funkcie z kontroléru, ktorá vracia dostupné pútka vo formáte json. `Type` obsahuje implicitne obsahuje „GET“, preto v našom prípade pri „POST“, teda odosielaní požiadavku musíme definovať aj `dataType`, ktorý definuje formát v akom očakávame odpoveď. V prípade úspešného prijatia požiadavku prebehne funkcia s argumentom `response`, ktorý zastupuje odpoveď serveru. Ten získava dáta z databázy, ktorá bola predtým pripojená. Funkcia následne používa for cyklus, kde z atribútu `isdefault` zisťuje, či je veľkosť pútka implicitná. V prípade, že áno, získa jeho cenu a zavolá funkciu `calculatePrice()`, ktorá následne vypočíta cenu celej kabelky. Funkcia taktiež v tomto prípade označí variantu ako aktívnu voľbu. Ak veľkosť pútka nie je implicitná, funkcia možnosť iba vypíše.

```
1  $.ajax({
2      url: '',
3      type: "POST",
4      dataType: 'json',
5      success: function (response) {
6          for (var i~= 0; i~< response.length; i++) {
7              if (response[i].isdefault == 't') {
8                  calculatedPrice.hangerSizePrice = parseInt(
                       response[i].price);
9                  calculatePrice();
10                 $("#hangerSizes").append('' + response[i].name
                       + '' + response[i].price + 'Kc');
11             } else {
12                 $("#hangerSizes").append('' + response[i].name
                       + '' + response[i].price + 'Kc');
13             }
14         }
15     },
16     error: function (xhr, ajaxOptions, thrownError) {
17         alert(xhr.status);
```

```
18          console.log(xhr.responseText);
19          alert(thrownError);
20      }
21  });
```

Taktiež môže nastať situácia, keď k úspešnému prijatiu požiadavku nedôjde. Vtedy sa zavolá funkcia, ktorá musí mať 3 argumenty, jqXHR objekt, reťazec popisujúci typ chyby, ktorý nastal a doplnkovú výnimku. Často sa môžeme stretnúť s referenciou `this`. Tá najčastejšie odkazuje na objekt v kontexte, ktorý je posunutý do $.ajax v nastaveniach. V prípade, že kontext nie je špecifikovaný, referencuje sa priamo do nastavení Ajax (api.jquery.com, 2016).

4.8 Vlastný šablónovací systém

V rámci implementácie bol riešený problém, ako efektívne zobrazovať výsledky v rámci pohľadov, nakoľko framework CodeIgniter neobsahuje žiaden šablónovací systém. To poskytuje istú voľnosť a zobrazovanie tak môžeme ľahšie prispôsobiť vlastným potrebám. Najlepším riešením zobrazovania je príprava rozloženia alebo dynamickej šablóny. V našom prípade je výhodnejšie využitie šablóny, nakoľko nie je potrebné načítavať celú stránku, hlavička a pätička zostanú nezmenené a načítajú sa iba jej stredné časti.

Pre lepšiu prácu s načítavaním jednotlivých častí bola stránka rozdelená na hlavičku, pätu, ľavú časť a stred. Základným kameňom šablóny je `MY_Controller.php`, ktorý definuje premenné `template` a `data` ako polia. Funkcia `layout` pridáva do template jednotlivé časti stránky a volá pohľady s tým súvisiace. Na záver privolá pohľad `layout/index` a predá mu dáta z novovytvorenej šablóny. Ďalšia časť šablóny sa nachádza v pohľadoch, kde bol vytvorený priečinok layout. V ňom môžeme nájsť php súbory footer, header, header_logged a index. Súbor header vykresľuje hlavičku pre neprihláseného užívateľa, zobrazuje mu možnosť prihlásenia a registrácie. Obsahuje HTML formuláre na prihlásenie a registráciu a taktiež javascripty, ktoré ich ošetrujú. Header_logged zobrazuje košík užívateľa, načítava jeho obsah a zobrazuje meno prihláseného používateľa. Taktiež obsahuje kontrolu prihlásenia admina. V prípade, že rozpozná prihlásenie administrátora, zobrazí v hlavičke správu produktov, užívateľov a objednávok. Index spúšťa všetky potrebné skripty na fungovanie použitých rozšírení ako toastr či validator. V home zase nájdeme načítanie všetkých produktov a ošetrenie zobrazovania detailu v prípade prihláseného a neprihláseného užívateľa.

```
1  class MY_Controller extends CI_Controller {
2
3      //set the class variable.
4      var $template = array();
5      private $data = array();
6
7      //Load layout
```

```
8     public function layout($viewData) {
9         // making temlate and send data to view.
10        $data = $viewData;
11        if ($this->authentication->is_loggedin()) {
12            // User is logged in
13            $this->load->model('user');
14            $user = $this->user->get_active_user($this->
                  authentication->read("username"));
15            $data["firstname"] = $user["first_name"];
16            $data["lastname"] = $user["last_name"];
17            $data["idUser"] = $user["id"];
18            $this->template['header'] = $this->load->view('
                  layout/header_loggedin', $data, true);
19        } else {
20            // User is NOT logged in
21            $this->template['header'] = $this->load->view('
                  layout/header', $data, true);
22        }
23        $this->template['content'] = $this->load->view($this->
              content, $this->data, true);
24        $this->template['footer'] = $this->load->view('layout/
              footer', $this->data, true);
25        $this->load->view('layout/index', $this->template);
26    }
```

5 Diskusia a záver

Cieľom práce bolo vytvoriť riešenie e-shopu s konfiguračnou aplikáciou, podporujúcou užívateľskú kustomizáciu produktov. Po analýze existujúcich riešení daného problému som sa rozhodla o vlastnú implementáciu riešenia, nakoľko požiadavky zadávateľky boli veľmi špecifické, hlavne čo sa týka možností budúcich rozšírení.

Pri získavaní teoretických znalostí boli využité prevažne anglické knihy a webové zdroje. Cieľom kapitoly s názvom Prehľad literatúry bolo oboznámenie sa s problematikou produktovej kustomizácie, procesu kustomizácie produktov, rozšírenému modelu kustomizácie produktu či vývoja moderných webových aplikácií. Pozornosť bola venovaná aj prieskumu existujúcich riešení na trhu.

Cieľom kapitoly Materiály a metodika bolo popísanie spôsobu riešenia problému implementácie aplikácie a oboznámenie sa s nástrojmi, ktoré boli pri implementácii využívané. Znalosti boli získané z oblasti UML diagramov, fungovania MVC architektúry, frameworku CodeIgniter, ktorý architektúru aplikuje a ďalších programových nástrojov, ako Javascript alebo Bootstrap. Ako zdroje informácií slúžili hlavne webové stránky vývojárov jednotlivých nástrojov.

Špecifikácia funkčných a nefunkčných požiadaviek, tvorba diagramu prípadov použitia a entitno-relačný model štruktúry ukladania dát boli súčasťou výsledkov práce. Ďalej bolo navrhnuté užívateľské rozhranie a vytvorené boli drôtové modely jednotlivých rozhraní. V druhej časti kapitoly bola opísaná štruktúra aplikácie. Na záver bola uvedená ukážka konkrétnych riešení implementácie, a to implementácie ukladania a zmeny farieb v konfiguračnej aplikácii, spôsob získavania dát zo serveru pomocou Ajaxu a nakoniec popis implementácie vlastného šablónovacieho systému. Hlavným prínosom aplikácie je výrazné zjednodušenie zadávania špecifických dizajnových požiadaviek na vzhľad kabelky od zákazníka ku výrobcovi. Vizualizáciou sa predchádza nesprávnemu pochopeniu predstáv zákazníka a taktiež je podporená jeho predstavivosť. Možnosť kustomizácie produktov priamo v e-shope navyše poskytuje konkurenčnú výhodu, nakoľko ide o ojedinelé rozšírenie v radoch konkurencie.

Nevýhodou aplikácie je absencia zobrazovania zmien vo veľkosti kabelky a typu pútka. Táto funkcionalita nebola naimplementovaná, nakoľko neboli k dispozícii fotografie produktov (ani reálne produkty, ktoré by bolo možné nafotiť), ktoré by umožňovali použitie obmeny iba zmenených modulov. V rámci budúcich rozšírení aplikácie je plánovaná možnosť voľby typu materiálu a dizajnu vrchnej časti kabelky. Taktiež je cieľom implementácia funkcionality zobrazovania všetkých úprav, nielen zmeny farby. V takom prípade bude zobrazovanie upravené a namiesto zmeny celej fotky bude zmenený iba daný modul, resp. časť fotografie produktu. Neskôr by mohla byť aplikácia rozšírená o možnosť ukladania návrhu, galérie zákazníckych návrhov či zdieľania na sociálnych sieťach. Využité by taktiež mohli byť 3D modely, ktoré by zaručili ešte reálnejšie zobrazovanie výsledného produktu.

Na záver je možné uviesť, že cieľ práce bol naplnený. Aplikácia, ktorá je súčasťou implementovaného riešenia, poskytuje možnosť užívateľskej kustomizácie produktu a výrazne zjednodušuje definovanie a ukladanie požiadaviek zákazníkov.

6 Literatúra

ADDISON, T., VALLABH, S. *Controlling Software Project Risks – an Empirical Study of Methods used by Experienced Project Managers*. Článok v žurnále: In proceedings of the South African Institute For Computer Scientists and Information Technologists, 2002. 128-140 s. ISBN 1-58113-596-3.

APACHE.ORG *Apache HTTP Server Project* [online]. Dokument vo formáte HTML. The Apache Software Foundation. 2016, [cit. 15.12.2016]. Dostupné z: `https://httpd.apache.org`.

API.JQUERY.COM *API Documentation, jQuery.ajax()* [online]. Dokument vo formáte HTML. The jQuery Foundation. 2016, [cit. 17.12.2016]. Dostupné z: `http://api.jquery.com/jquery.ajax/`.

BLECKER, T., FRIEDRICH, G., KALUZA, B., ABDELKAFI, N., KREULTER, G. *Information and Management Systems for Product Customization*. 1. vydanie, New York, NY: Springer Science & Business Media, 2004. 269 s. ISBN 978-03-872-3347-5.

CHEN, S., WANG, Y., TSENG, M.M. *Mass Customization as a Collaborative Engineering Effort* [online]. Dokument vo formáte PDF. The Hong Kong University of Science & Technology, Hong Kong. 2009, [cit. 5.12.2016]. 28 s. Dostupné z: `http://www3.ntu.edu.sg/home/songlin/Attachments/IJCE09.pdf`.

CHROME.COM *MVC Architecture* [online]. Dokument vo formáte HTML. Google. 2016, [cit. 15.12.2016]. Dostupné z: `https://developer.chrome.com/apps/app_frameworks`.

CODEIGNITER.COM *CodeIgniter Overview* [online]. Dokumenty vo formáte HTML. British Columbia Institute of Technology. 2016, [cit. 13.12.2016]. Dostupné z: `https://www.codeigniter.com/userguide3/overview/`.

FLANAGAN, D. *JavaScript: The Definitive Guide*. 5. vydanie, Sepastopol, CA: O'Reilly Media, Inc., 2006. 994 s. ISBN 978-05-961-0199-2.

FLUID.COM *What we do – product customization* [online]. Dokument vo formáte HTML. Fluid, Inc. 2015, [cit. 11.12.2016]. Dostupné z: `https://www.fluid.com/what-we-do/product-customization`.

GALITZ, W.O. *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques Wiley Desktop Editions*. 3. vydanie, Indianapolis, IN: John Wiley & Sons., 2007. 857 s. ISBN 978-04-7014-6224.

GETBOOTSTRAP.COM *About Bootstrap* [online]. Dokument vo formáte HTML. 2016, [cit. 13.12.2016]. Dostupné z: `http://getbootstrap.com/about/`.

GITHUB.COM *CodeSeven/toastr* [online]. Dokument vo formáte HTML. GitHub, Inc. 2016, [cit. 15.12.2016]. Dostupné z: `https://github.com/CodeSeven/toastr`.

HOWARD, M. *Bezpečný kód: techniky a strategie tvorby bezpečných webových aplikací*. Brno: Computer Press., 2008. 895 s. ISBN 978-80-251-2050-7.

HVAM, L., MORTENSEN, N.H., RIIS, J. *Product Customization*. 1. vydanie, Berlín: Springer Science & Business Media., 2008. 283 s. ISBN 978-35-407-1449-1.

IŠOVÁ H. *Profil uživatele Helena Išová* [online]. Dokument vo formáte HTML. Fler - kreativní svět. 2011, [cit. 5.12.2016] Dostupné z: `https://www.fler.cz/helena-isova`.

JQUERY.COM *jQuery Website* [online]. Dokument vo formáte HTML. The jQuery Foundation. 2016, [cit. 15.12.2016]. Dostupné z: `https://jquery.com`.

KOHAN B. *Guide to Web Application Development* [online]. Dokument vo formáte HTML. Comentum. 2016, [cit. 12.12.2016]. Dostupné z: `http://www.comentum.com/guide-to-web-application-development`.

KOHAN B. *Web Application Development Process* [online]. Dokument vo formáte HTML. Comentum. 2016, [cit. 12.12.2016]. Dostupné z: `http://www.comentum.com/web-application-development-process`.

MACINTYRE, D. *PHP: the good parts*. 1. vydanie, Sebastopol, CA: O'Reily Media, Inc., 2010. 156 s. ISBN 978-0-59680-437-4.

MILOPLE.COM *Personalized products* [online]. Dokument vo formáte HTML. Milople.com. 2016, [cit. 11.12.2016]. Dostupné z: `https://www.milople.com/magento-extensions/personalized-products.html`.

MYSQL.COM *MySQL Engineering Blogs* [online]. Dokument vo formáte HTML. Oracle Corporation. 2016, [cit. 11.12.2016]. Dostupné z: `http://dev.mysql.com`.

NARAMORE, E. *Vytváříme webové aplikace v PHP5, MySQL a Apache*. 1. vydanie, Brno: Computer Press, 2006. 813 s. ISBN 80-251-1073-7.

PHP.NET *Versions of PHP* [online]. Dokument vo formáte HTML. php.net. 2016, [cit. 13.12.2016]. Dostupné z: `http://php.net/downloads.php`.

PILONE, D., PITMAN, N. *UML 2.0 in a Nutshell*. Sebastopol, CA: O'Reily Media, Inc., 2005. 216 s. ISBN 978-0-596-00795-9.

PINE, B.J., GILMORE, J.H. *The Experience Economy: Work is Theatre & Every Business a Stage*. 1. vydanie, Brighton, MA: Harvard Business Publishing., 1999. 254 s. ISBN 978-08-758-4819-8.

PORTER, M.E. *Competitive Advantage of Nations: Creating and Sustaining Superior Performance*. 2. vydanie, New York, NY: Simon and Schuster., 1995. 896 s. ISBN 978-06-848-4147-2.

PRODUCTCART.COM *ProductCart Configurator* [online]. Dokument vo formáte HTML. NetSource Commerce, Inc. 2016, [cit. 11.12.2016]. Dostupné z: http://www.productcart.com/feature-tour/.

REENSKAUG, T. *Trygve/MVC* [online]. Dokument vo formáte HTML. Hjemmesider ved Institutt for informatikk., 1995, [cit. 11.12.2016]. Dostupné z: http://heim.ifi.uio.no/trygver/themes/mvc/mvc-index.html.

SPAULDING, E., PERRY, C. *Business Insights - Making it personal: Rules for success in product customization* [online]. Dokument vo formáte PDF. Bain & Company. 2013, [cit. 11.12.2016]. 8 s. Dostupné z: http://www.bain.com/publications/articles/making-it-personal-rules-for-success-in-product-customization.aspx.

SPRING, M., DALRYMPLE, J.F. *Product customisation and manufacturing strategy* [online]. Žurnál vo formáte PDF. International Journal of Operations & Production Management, Vol. 20, No. 4. Bingley: MCB University Press. 2000, [cit. 2.12.2016]. 441-467 s. Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.551.9661&rep=rep1&type=pdf.

UML.ORG *What is UML* [online]. Dokument vo formáte HTML. Object Management Group, Inc. 2005, [cit. 13.12.2016]. Dostupné z: http://www.uml.org/what-is-uml.htm.

UPTON, D. *Codeigniter for Rapid Php Application Development*. 1. vydanie, Birmingham, U.K: Packt Publishing Ltd., 1995. 220 s. ISBN 1-84719-175-4.

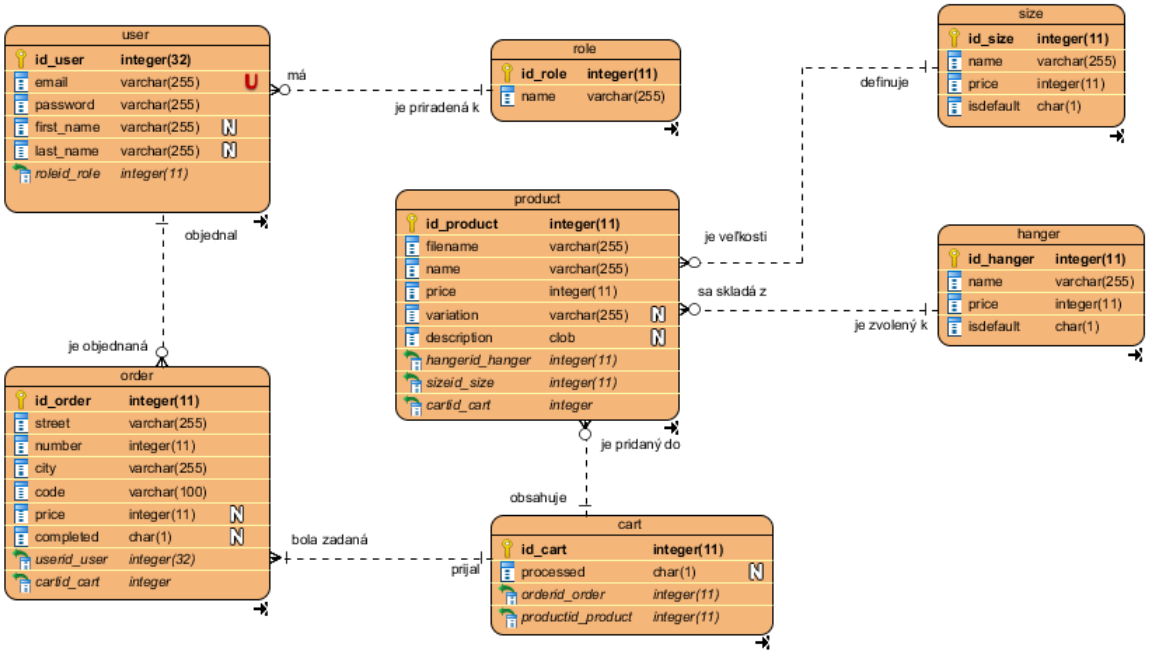
VISUAL-PARADIGM.COM *Entity Relationship Diagram* [online]. Dokumenty vo formáte HTML. Visual Paradigm. 2016, [cit. 16.12.2016]. Dostupné z: https://www.visual-paradigm.com/VPGallery/datamodeling/EntityRelationshipDiagram.html.

VISUAL-PARADIGM.COM *UML Use case diagram* [online]. Dokumenty vo formáte HTML. Visual Paradigm. 2016, [cit. 16.12.2016]. Dostupné z: https://www.visual-paradigm.com/support/documents/vpuserguide.

W3SCHOOLS.COM *AJAX Introduction* [online]. Dokument vo formáte HTML. W3.CSS. 2016, [cit. 15.12.2016]. Dostupné z: http://www.w3schools.com/xml/ajax_intro.asp.

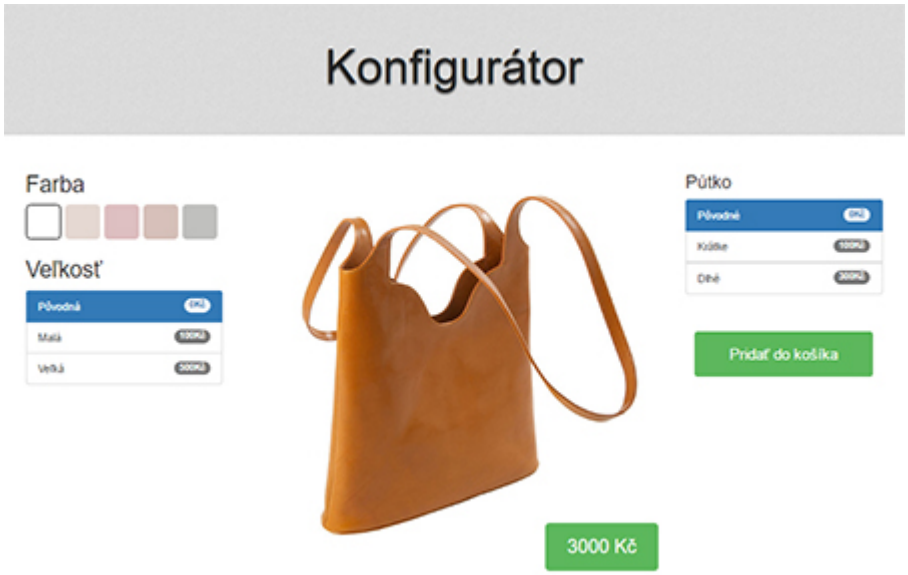
Prílohy

A Fyzický model databázy



Obrázok 9: Fyzický model databázy

B Ukážka prostredia konfiguračnej aplikácie



Obrázok 10: Prostredie konfiguračnej aplikácie