

Dokumentácia

Zadanie: Webshop

Flóra Emma Kaňuchová, Zsófia Gergely

Predmet: WTECH_B

11. 5. 2025

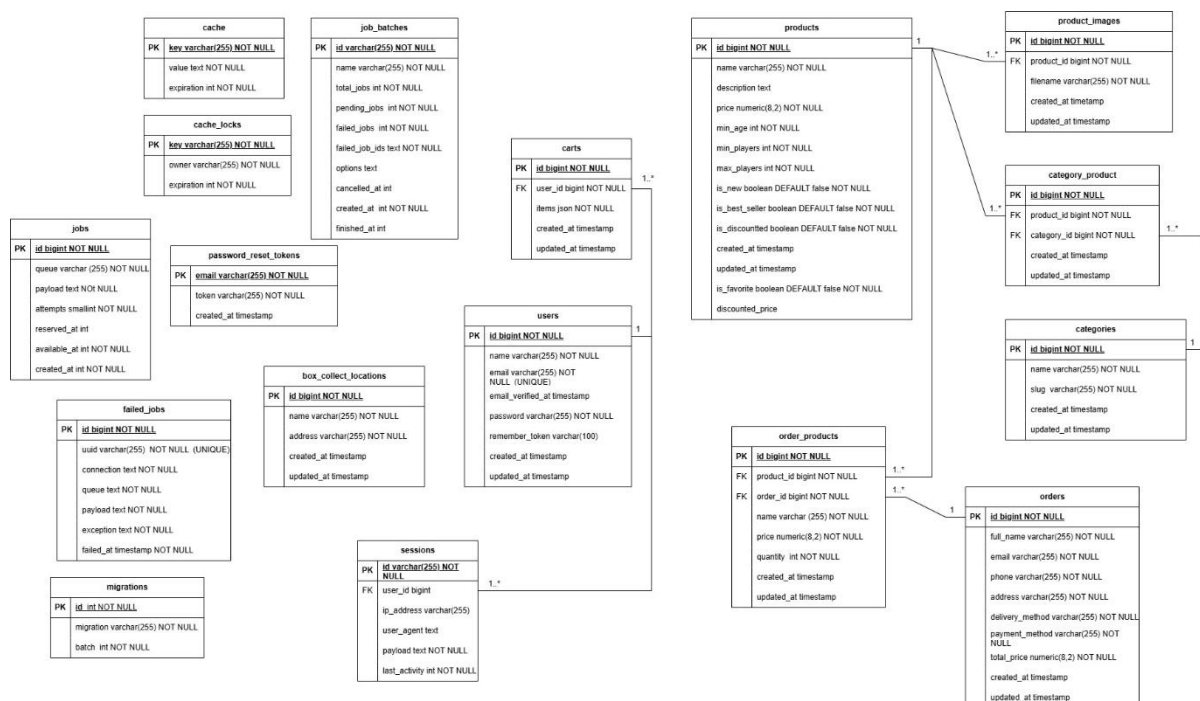
Stručný opis zadania:

V rámci školského projektu sme vytvorili webový obchod zameraný na spoločenské hry. Cieľom bolo navrhnuť a implementovať plne funkčný e-shop, ktorý umožňuje používateľom prezerať si ponuku hier, pridávať produkty do košíka a dokončiť objednávku.

Na vývoj sme použili framework Laravel, ktorý nám zabezpečil robustnú backendovú logiku a bezpečnú prácu s databázou. Pre dizajn a responzívne rozhranie sme využili Bootstrap, vďaka čomu je stránka prehľadná a použiteľná aj na mobilných zariadeniach.

Projekt nám pomohol lepšie pochopiť princípy webového vývoja a tímovej spolupráce.

Diagram fyzického dátového modelu:



V druhej fáze projektu sme výrazne upravili fyzický databázový model oproti prvej fáze. Hlavné zmeny zahŕňajú prechod na anglické názvy tabuliek a stĺpcov, integráciu štandardných Laravel tabuliek, optimalizáciu štruktúry pre lepšiu funkčnosť webshopu a rozšírenie modelu o nové funkcionality.

Systém využíva štandardné Laravel tabuľky ako: cache pre ukladanie dočasných cache dát, cache_locks pre zámky cache, migrations pre správu migračných súborov, jobs, job_batches a failed_jobs pre správu úloh vo fronte, password_reset_tokens pre správu tokenov na resetovanie hesla, sessions pre sledovanie relácií používateľov, users pre používateľské účty.

Okrem týchto systémových tabuliek bol model rozšírený o tabuľky potrebné pre fungovanie e-shopu: products, ktoré obsahujú informácie o produktoch vrátane cien, počtu hráčov, vekového obmedzenia a rôznych flagov ako bestseller, zľava či obľúbenosť, product_images pre správu

obrázkov produktov (vzťah 1:N k produktom), categories a prepojavacu tabuľku category_product pre podporu kategorizácie produktov (vzťah M:N), carts pre ukladanie položiek v košíkoch používateľov, orders a order_products pre správu objednávok a ich položiek (vzťah 1:N), box_collect_locations ako podpora pre výdajné miesta objednávok.

Medzi kľúčové zmeny patrí aj zavedenie atribútov ako is_discounted a discounted_price, ktoré slúžia na označenie akciových a zľavnených produktov pre jednoduchšie filtrovanie používateľom, a is_favorite, ktorý umožňuje používateľom ukladať si obľúbené produkty. Model taktiež zohľadňuje atribúty ako počet hráčov a vekové obmedzenie, čo zvyšuje presnosť filtrovania.

V priečinku webshop/database/schema sa nachádza vygenerovaný súbor schema_data_DDL.sql, ktorý bol vytvorený pomocou PgAdmin 4. Obsahuje našu databázu vo formáte DDL vrátane schémy aj dát.

Treba však poznamenať, že pri vytváraní databázy sme použili Laravel – všetky tabuľky, vzťahy a počiatočné údaje boli vytvorené pomocou migrácií, seederov a modelov.

Návrhové rozhodnutia:

Bootstrap:

Pomocou frameworku Bootstrap bolo jednoduché vytvoriť responzívne stránky. Využili sme predovšetkým 12-stĺpcový grid systém, ktorý nám umožnil rýchlo a efektívne navrhnuť rozloženie jednotlivých komponentov na stránke. Okrem toho Bootstrap výrazne urýchlil vývoj vďaka preddefinovaným triedam pre štylovanie.

Priame SQL výrazy v PHP (NavBarController, SideBarController):

V niektorých prípadoch sme museli použiť priame SQL výrazy priamo v PHP kóde. Hlavným dôvodom bolo správne filtrovanie a triedenie produktov podľa ceny, kde bolo potrebné zohľadniť zľavnenú cenu produktu (ak existuje), namiesto pôvodnej. Tento prístup zabezpečil, že napríklad pri filtrovaní produktov podľa cenového rozsahu, alebo pri triedení vzostupne/zostupne podľa ceny, bola zohľadnená práve aktuálna cena po zľave.

JavaScript:

Vlastné JavaScript súbory sme umiestnili do priečinka webshop/public/js. Tieto skripty zabezpečujú viaceré funkcie v rámci používateľského rozhrania. Používajú sa na overovanie údajov pri platbe, ako aj na kontrolu vyplnenia formulárových polí pri prihlasovaní a registrácii. Na domovskej stránke zabezpečujú dynamické zobrazovanie produktov v jednotlivých kategóriách: Akcie, Novinky a Best sellery. Na stránke detailu produktu umožňujú interaktívne prepínanie medzi obrázkami produktu.

Programátorské prostredie:

Projekt bol vyvíjaný na operačnom systéme Windows 11 s použitím vývojového balíka XAMPP, hoci na prednáškach bol odporúčaný WAMP. Ak by ste používali WAMP, nemal by s tým byť žiadny problém. Dôvod, prečo sme zvolili XAMPP, je ten, že jeden člen tímu s ním už mal predchádzajúce skúsenosti.

Na vývoj backendu sme použili Laravel, zatiaľ čo frontend bol postavený s využitím Bootstrapu. Projekt bol naprogramovaný pomocou jazykov PHP, HTML, CSS a JavaScript.

Ako databázu sme použili PostgreSQL, ktorú sme spravovali cez nástroj PgAdmin. Vývoj prebiehal v editore Visual Studio Code. Na správu verzií sme používali Git Bash a GitHub.

Stručný opis implementácie prípadov použitia:

Zmena množstva pre daný produkt:

V ProductController metóda `updateCartQuantity()` slúži na zmenu množstva konkrétneho produktu v nákupnom košíku: buď na jeho zvýšenie, alebo zníženie.

Priebeh metódy:

1. Najprv získa aktuálny obsah košíka zo session (`session()->get('cart', [])`).
2. Skontroluje, či sa v košíku nachádza daný produkt.
 - a. Ak sa požaduje zvýšiť množstvo (`increase`), produktová položka sa zvýši o 1.
 - b. Ak sa požaduje znížiť množstvo (`decrease`), overí sa, či je aktuálne množstvo väčšie ako 1 – a ak áno, množstvo sa zníži o 1 (takto sa zabráni tomu, aby množstvo kleslo pod 1).
3. Aktualizovaný košík sa opäť uloží do session (`session()->put('cart', $cart)`).
4. Zavolá sa súkromná metóda `saveCartToDatabase()`, ktorá zabezpečí, že aktuálny stav košíka sa uloží aj do databázy – ale len v prípade, že je používateľ prihlásený.

Prihlásenie/Odhlásenie:

V AuthController metódy `login()` a `logout()` slúžia na prihlásenie a odhlásenie používateľa v systéme pomocou e-mailovej adresy a hesla.

Priebeh metódy `login()`:

1. Z požiadavky sa získa dvojica údajov: email a password.
2. Pomocou `Auth::attempt()` sa overí, či tieto údaje zodpovedajú existujúcemu používateľovi.
3. Ak je prihlásenie úspešné: Získa sa aktuálny používateľ a jeho košík zo session.
4. Následne sa z databázy načíta aj jeho uložený košík (Cart).
5. Tieto dva košíky sa zlúčia:
 - a. Ak produkt už existuje v databázovom košíku, pripočíta sa množstvo zo session.
 - b. Inak sa produkt pridá do zoznamu.
6. Výsledný košík sa uloží do databázy a zároveň aj do session.
7. Používateľ je presmerovaný na stránku s poďakovaním.
8. Ak je prihlásenie neúspešné, používateľ zostáva na stránke.

Priebeh metódy logout():

1. Z pamäte (session) sa odstráni obsah nákupného košíka pomocou session()->forget('cart').
2. Používateľ sa odhlási pomocou Auth::logout().
3. Nakoniec sa používateľ presmeruje na domovskú stránku (/).

Vyhľadávanie:

SearchController slúži na vyhľadávanie produktov podľa zadaného reťazca. Obsahuje dve metódy: súkromnú metódu na normalizáciu textu a verejnú metódu, ktorá spracováva samotné vyhľadávanie.

Metóda removeAccents(\$string):

- Typ: súkromná metóda (private)
- Účel: Odstraňuje diakritiku zo zadaného reťazca.
- Využitie: Používa sa na porovnávanie textov bez ohľadu na akcenty (napr. „ž“ sa premení na „z“).

index(Request \$request):

- Typ: verejná metóda (public)
- Účel: Spracúva vyhľadávací dopyt od používateľa a vracia produkty, ktoré zodpovedajú zadanej fráze.

Postup:

1. Metóda načíta reťazec hľadania (query), ako aj voliteľné filtre ako min_price, max_price, vekova_kategoria, hracov a sort.
2. Hľadaný výraz sa prevedie na malé písmená a odstránia sa diakritické znamienka pre zabezpečenie tolerantného vyhľadávania.
3. Všetky produkty sa načítajú spolu s ich obrázkami pomocou Product::with('images')->get().
4. Kolekcia produktov sa prefiltruje podľa týchto podmienok: názov produktu obsahuje hľadaný výraz (bez diakritiky).
5. Zoradenie výsledkov: Výsledná kolekcia sa zoradí podľa zvoleného parametra (sort), ktorý môže byť: price_asc: od najlacnejšieho, price_desc: od najdrahšieho, asc: podľa názvu A-Z (predvolené), desc: podľa názvu Z-A.
6. Stránkovanie výsledkov: Výsledky sa rozdelia po 12 kusoch na stranu pomocou LengthAwarePaginator.
7. Zobrazenie pohľadu: Vracia sa view shop.blade.php s produktmi, názvom kategórie (napr. "Výsledky hľadania pre: „nejeká hra“") a hodnotami filtrov.

Pridanie produktu do košíka:

V ProductController metóda addToCart() pridáva produkt so zadaným ID do nákupného košíka používateľa, pričom rozlišuje medzi prihláseným a neprihláseným používateľom.

Postup:

1. Získanie produktu:
 - a. Na začiatku sa vyhladá produkt podľa ID. Ak sa nenájde, operácia skončí s chybou.
 - b. Zistí sa, či je produkt v zľave. Ak áno, použije sa zľavnená cena, inak bežná.
2. Neprihlásený používateľ:

- a. Košík sa ukladá do session.
 - b. Ak už produkt v košíku existuje, navýši sa jeho množstvo o jeden kus.
 - c. Ak produkt ešte nie je v košíku, vytvorí sa nová položka s jeho údajmi a množstvom 1.
3. Prihlásený používateľ:
 - a. Košík sa ukladá do databázy (tabuľka carts), pričom údaje sa uchovávajú vo formáte JSON.
 - b. Ak už má používateľ existujúci košík, načítajú sa jeho položky.
 - c. Produkt sa buď pridá ako nový, alebo sa navýši jeho množstvo.
 - d. Aktuálny stav košíka sa následne uloží aj do session, aby sa zabezpečila konzistentnosť medzi databázou a prehliadačom.

Stránkovanie:

Uskutoční sa v NavBarController, ProductController, SearchController, SideBarController. Stránkovanie v aplikácii zabezpečuje rozdelenie produktov do stránok po 12 položiek.

Na stránkovanie sme použili komponent Bootstrap pagination, ktorý bol začlenený cez `{{ $products->links('pagination::bootstrap-5') }}` v `shop.blade.php`.

Aby sme odstránili predvolený text (informáciu o zvyšných počtoch strán alebo výsledkov), ktorý sa pri stránkovaní Laravelom často zobrazuje, bolo potrebné upraviť predvolenú šablónu stránkovania. To sme dosiahli publikovaním pagination šablón, čím sme získali možnosť opraviť výstup v zložke `resources/views/vendor/pagination`.

Stránkovanie na úrovni databázy:

V kontroléroch NavBarController a SideBarController sa stránkovanie realizuje priamo pri načítaní údajov z databázy. Používa sa na to funkcia, ktorá automaticky rozdelí produkty na stránky a zároveň si uchová parametre z URL adresy, ako sú filtre alebo zoradenie.

Stránkovanie v pamäti (na úrovni kolekcie):

V ProductController a SearchController sa produkty najprv všetky načítajú a až potom sa filtrujú v aplikácii. Po aplikovaní filtrov sa výsledky rozdelia na stránky manuálne. Takéto riešenie sa používa tam, kde sa pracuje s komplexnejším spracovaním údajov, ktoré nie je možné alebo efektívne vyriešiť priamo v databáze.

Zobrazenie stránkovania vo výstupe:

Zohľadňuje všetky aktuálne filtre, zoradenie a ďalšie parametre, aby používateľ pri prechádzaní nestratil kontext svojho výberu.

Základné filtrovanie:

Uskutoční sa v ProductController, NavBarController, SideBarController. V aplikácii sú dostupné viaceré filtre, ktoré fungujú jednotne naprieč viacerými stránkami a kategóriami.

1. Veková kategória:

Používateľ môže určiť maximálny odporúčaný vek pre hru, ktorý ho zaujíma. Aplikácia následne zobrazí všetky hry vhodné pre deti alebo osoby s nižším vekom ako zadaná hodnota.
2. Počet hráčov:

Filtrovanie umožňuje vybrať minimálny počet hráčov, ktorí sa môžu zapojiť do hry. Zobrazia sa iba tie hry, ktoré sú navrhnuté pre daný počet alebo viac hráčov.
3. Cena (od - do):

Používateľ si môže nastaviť minimálnu a/alebo maximálnu cenu hry. V prípade zľavnených produktov sa pri filtrovaní zohľadňuje akciová cena, ak je dostupná, inak sa berie do úvahy bežná cena.

4. Zoradenie výsledkov:

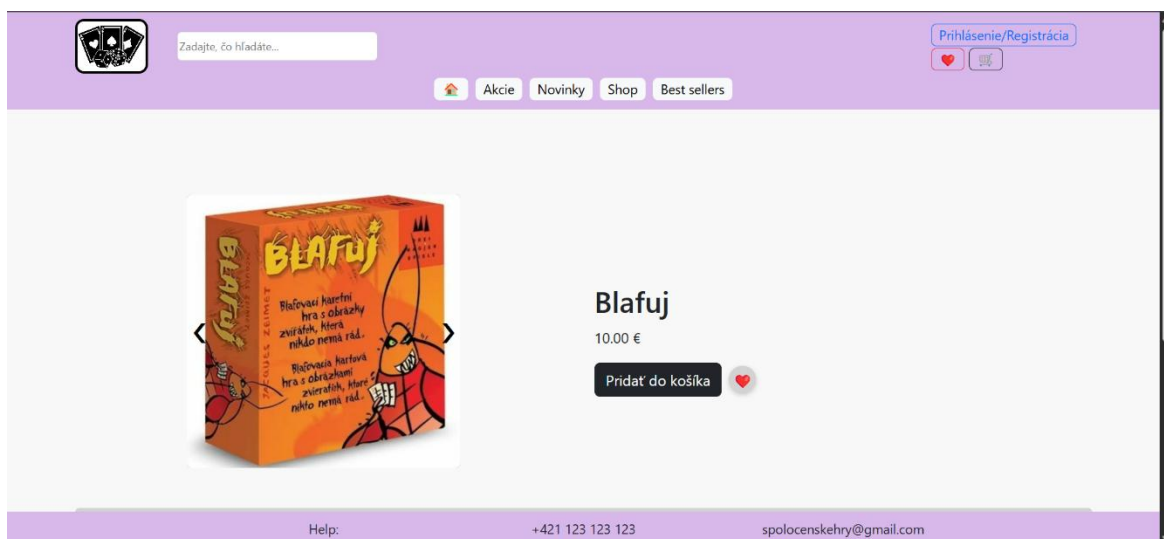
Okrem filtrov možno produkty aj zoradiť – podľa názvu alebo ceny, vzostupne alebo zostupne.

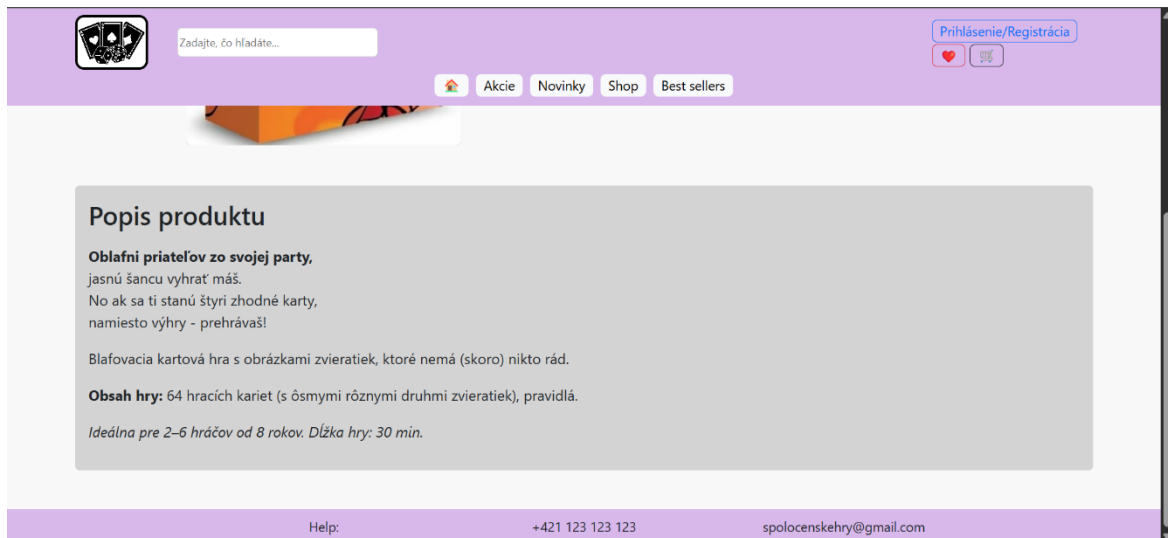
Zachovanie filtrov pri stránkovaní:

Pri prechádzaní medzi stránkami sú filtre automaticky uchované v adrese (URL), aby sa zabezpečila konzistentnosť výsledkov. Ak ProductSeeder bol spustený len raz, tak môžete si to vyskúšať v shop (v NavBar je na to button) tak, že nastavíte filter: cena od 1 do 25, maximálna veková kategória 10 rokov a minimálny počet hráčov 4. Hry, ako Activity, Fabio, Iq Link, Meme, Monopoly, Pexesohm, Puzzle Kvet a Puzzle Lalia nebudú zobrazené, pretože nespĺnia aspoň jeden z daných kritérií.

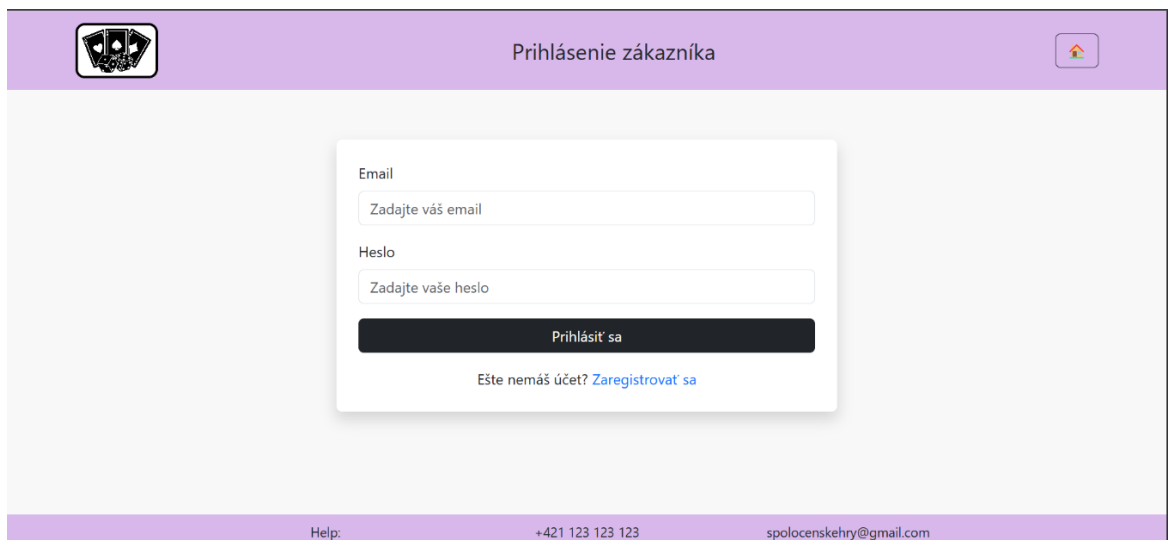
Snímky obrazoviek:

Detail produktu:

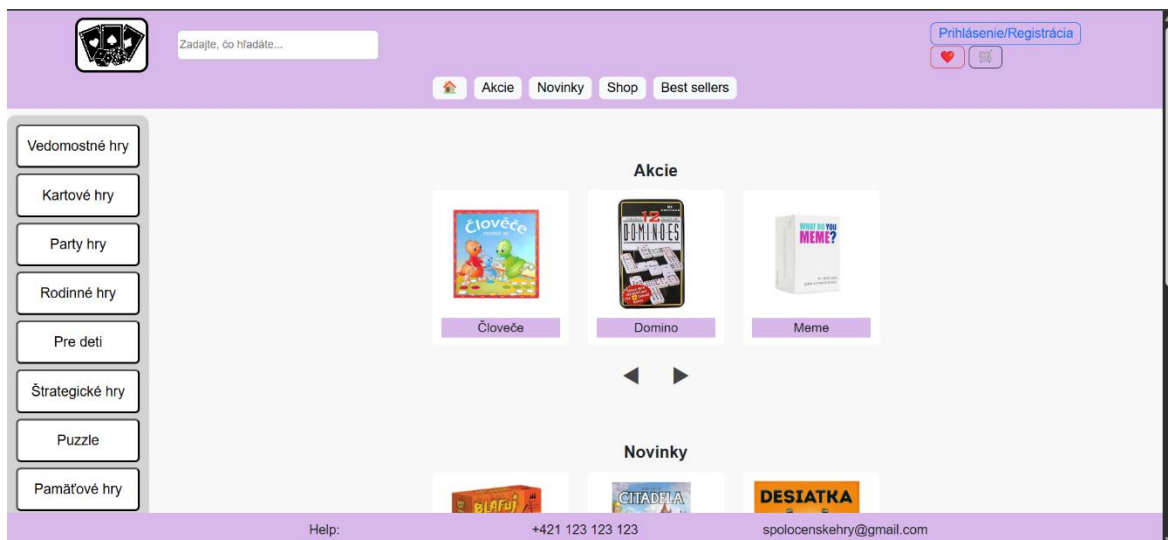


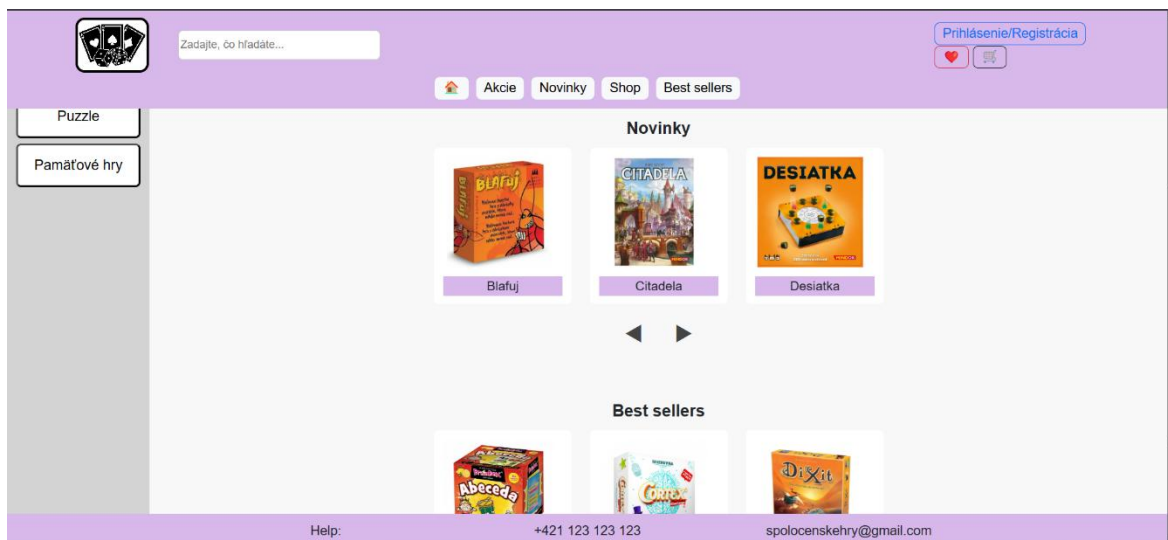


Prihlásenie:

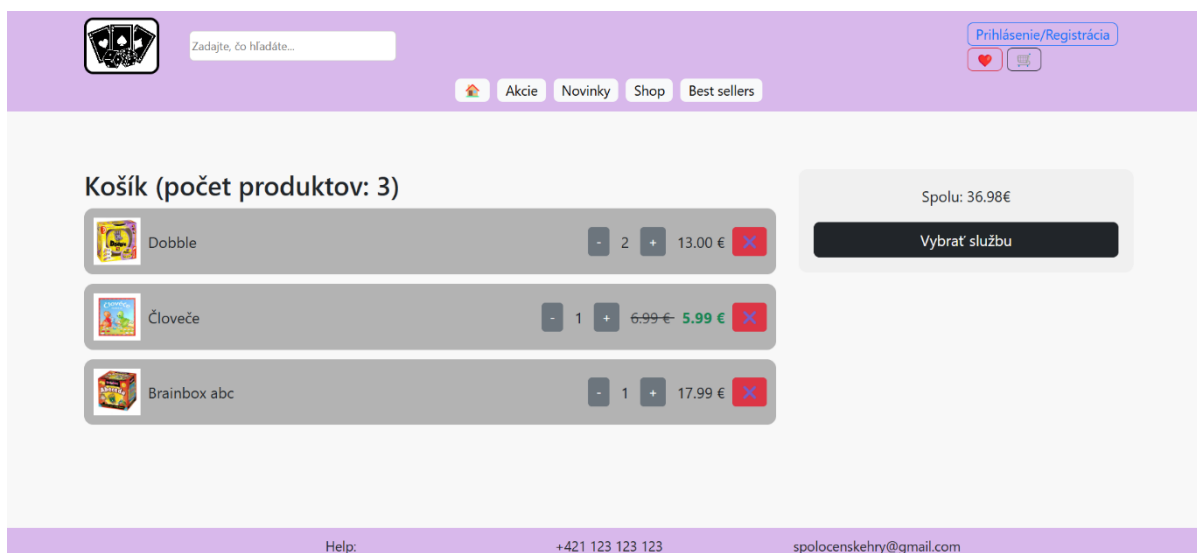


Homepage:





Nákupný košík s vloženým produktom:



Dodatočne pridané funkcionality:

Validácia formulárov cez JavaScript:

Skripty boxcollect.js, kurierska.js, kontrola.js a časť platba.js slúžia na overovanie správnosti vstupov vo formulároch. Kontrolujú, či sú všetky povinné polia vyplnené a či majú správny formát. Ak používateľ napríklad ponechá niektoré pole prázdne, systém zobrazí chybové hlásenie.

Filtrovanie a triedenie podľa ceny zohľadňuje zľavnené ceny:

Pri filtrovaní produktov podľa ceny a pri zoradení sa zohľadňuje zľavnená cena (discounted_price), ak existuje. Ak produkt nie je v zľave, použije sa jeho štandardná cena.

Oblíbené produkty:

Použivatel si může přidat produkty mezi oblíbené kliknutím na ikonu srdce při produktu. Tieto produkty sa následne zobrazia v sekcii „Oblíbené produkty“, ktorú možno otvoriť kliknutím na ikonu srdca v navigačnom paneli (NavBar). Opätovným kliknutím na srdce sa produkt zo zoznamu oblíbených odstráni.

Používateľská príručka:

- Môžete naklonovať náš repozitár príkazom: “git clone <https://github.com/Zsofkaaa/WTECH>”.
- Musíte mať nainštalovaný aspoň PHP 8.2 a v súbore php.ini odkomentované rozšírenia: fileinfo, gd, pgsql, pdo_pgsql.
- Spustíte príkaz: “composer install”. (možno budete potrebovať aj “npm install”)
- Vytvorte súbor .env a podľa súboru .env.example správne nastavte premenné DB_DATABASE, DB_USERNAME a DB_PASSWORD.
- Ďalej musíte vygenerovať kľúč príkazom: “php artisan key:generate”.
- V editore, ktorý používate na prácu s PostgreSQL, vytvorte databázu s rovnakým názvom, aký ste nastavili v súbore .env. Potom spustíte príkazy: “php artisan migrate” a “php artisan db:seed”.
- Ak by ste náhodou spustili seedery viackrát a údaje v databáze by boli duplikované, týmto príkazom viete ich znova načítať bez duplikácií: „php artisan migrate:fresh --seed“.
- (Keby ste chceli viac produktov, napríklad na kontrolovanie stránkovania, tak viete duplikovať iba produkty príkazom: „php artisan db:seed --class=ProductSeeder“.)
- Nakoniec môžete spustiť webový obchod na localhoste (127.0.0.1:8000) príkazom: “php artisan serve”.
- Keby ste chceli, aby aplikácia bola dostupná aj z telefónu alebo iného zariadenia, tak môžete používať príkaz: „php artisan serve --host=0.0.0.0 --port=8000“. Musíte zistiť aj svoju IP adresu („ipconfig“). V telefóne, alebo v inom zariadení otvorte prehliadač a zadajte IP adresu a port počítača. Telefón aj počítač musí byť pripojený na tú istú Wi-Fi sieť.