

Веб апликација за електронско купување на балони со Spring Framework

Изработиле:

Бојан Дикиќ 115020

Дејан Арсиќ 115005

Мила Јовановска 115032

Содржина:

**1.Опис на проектот4**

1.1 Бизнис идеја...................................................................................................................4

1.2 Бизнис модел.................................................................................................................4

**2. Техничка спецификација5**

2.1 Функционалности..........................................................................................................5

2.2 Упатство за поставување на проектот..........................................................................5

2.3 Опис на бизнис слој.......................................................................................................6

2.4 Опис на базата11

1. Опис на проектот
   1. Бизнис идеја

Нашата бизнис идеја произлегува од потребата за посебно специјализирано место каде што секој корисник ќе може да пронајде соодветен балон за секоја пригода. Подарување на балон или пак, декорација со балони е сигурен и проверен начин како да донесеме среќа на луѓето околу нас. Идејата за овој сајт, исто како што гласи и слоганот на веб страната е да можеме на луѓето да им пружиме мало парче среќа. Достапен во секое време, овој *e-commerce* сајт е најдобар начин за продажба на конкретен производ (во овој случај, балони) на поширока група на луѓе низ целиот свет.

* 1. Бизнис модел

Оваа веб страна нуди олеснување и намалување на времето на чекање за плаќање како и голем избор на балони и нивно лесно наоѓање. Предноста е во тоа што, доколку што знаеме што сакаме, тогаш оваа веб страна ги има сите видови на балони на едно место. Модел на наша заработка е продажбата на балони на нашите корисници. Првин започнуваме со покривање на пазарот во Македонија поради тоа што засега не постои специјализиран сајт за продажба на балони во нашата земја. Конкурентски компании се малку застапени во овој регион, па така конкуренцијата би била мала и доколку сајтот нуди услуги пошироко од Македонија. Што се однесува до организацијата зад апликацијата се наоѓа амбициозен тим којшто верува во успешноста и профитабилноста на проектот. Веб страната ќе биде достапна во секое време и секој корисник ќе може да ги прегледува производите без да мора да направи профил. Веруваме дека овој сајт ќе биде прифатен од сите, а најмногу од младата популација пред се поради едноставноста на самиот сајт, лесниот кориснички интерфејс и заштедата на време.

1. Техничка спецификација
   1. Функционалности Корисникот ќе има можност за:

* Преглед на балони
* Преглед на категории
* Преглед на балони по категорија
* Детален преглед на селектиран балон
* Додавање на балони во купувачка кошничка
* Уредување на купувачка кошничка
* Креирање на нарачка
* Наплата
* Најава и регистрација
* Пребарување на балони
* Преглед на балони на промоција
* Pin it опција за балоните
* Верификација на кориснички профил преку e-mail
* Информации за извршени нарачки на e-mail

Администраторот ќе има можност за:

* Најава/одјава
* Преглед и уредување на категории
* Преглед и уредување на производи
* Уредување на нарачки
* Преглед на нарачки
* Промена на статус на нарачки
* Додавање на категории/производи
* Додавање/отстранување на производи на промоција
  1. Упатство за поставување на проектот

Проектот треба да се симне од GitHub репозиториум преку следниот линк: <https://github.com/Dikic/balloonshop> .

Проектот треба да се импортира во Eclipse преку наредбата Import-> Existing Projects Into Workspace и потоа да се селектира симнатиот проект.

Потоа треба да се симне и да се инсталира Apache Tomcat Server. Упатство како да се направи тоа е дадено на следниот линк <http://www.eclipse.org/webtools/jst/components/ws/1.5/tutorials/InstallTomcat/InstallTomcat.html>

Во директориумот balloonshop\src\main\resources постои датотеката db.properties кој е конфигурациски фајл на проектот. Полето db.url треба да се смени во зависност од тоа со која база се работи. Доколку работите со локална база, тогаш полето не треба да се менува. Соодветно треба да се сменат полињата db.username и db.password соодветно.

Во фајлот AppConfig.java кој се наоѓа во balloonshop\src\main\java\mk\ukim\finki\emk\balloonshop треба да се промени локацијата за местото каде што се чуваат сликите. Соодветно треба да се промени вредноста на променливата UPLOAD\_PATH.

Ние работевме со xampp и со локална mysql база на податоци. Во директориумот balloonshop\src\main\resources се наоѓа датотеката balloonshop.sql со која се полни базата. Потребно е да се креира шема со име “balloonshop”, односно името треба да одговара со url-то. Табелите се креираат со стартување на проектот. За логирање веќе постои креиран администратор со корисничко име „admin“ и лозинка „admin“.

* 1. Опис на бизнис слој

|  |  |  |  |
| --- | --- | --- | --- |
| Име на класата | Опис | Функции | Опис на функциите |
| AdminController | Контролер за администраторскиот дел од страниците. Одоворен е за поврзување на моделот со соодветниот поглед кој се пристапува. | adminHome() | Го дава погледот за почетната страна на администраторот |
| usersGet() | GET метода која ги враќа сите корисници |
| deleteUser(@RequestParam int id) | Го брише корисникот со зададено ID |
| addUser(@ModelAttribute User user) | Додава корисник во базата |
| productsGet() | Ги дава сите производи |
| deleteProduct(@RequestParam int id) | Го брише производот со зададено ID |
| addProduct() | Додава производ во базата |
| categoriesGet() | Ги дава сите категории |
| deleteCategory(@RequestParam int id) | Ја брише категоријата со зададено ID |
| addProduct(@ModelAttribute Category category) | Додава продукт во категорија |
| editCategoryForm(@PathVariable int categoryId) | Го дава погледот за уредувае на категории |
| editUserForm(@PathVariable int userId) | Го дава погледот за уредување на корисници |
| BalloonshopController | Контролер за корисничкиот дел од страниците. Одговорен е за поврзување на страниците кои се пристапуваат од страна на корисниците со соодветните модели кои ја содржат логиката зад секој поглед. | getProductCount(HttpSession session) | Враќа број на производи во кошничка |
| getCategories() | Враќа листа од категории |
| getOnPromotions() | Враќа листа од производи на промоција |
| loginPost(@ModelAttribute User user, HttpSession session) | Го дава модалот за најава |
| signinPost(@ModelAttribute User user, HttpSession session) | Го дава модалот за креирање account |
| signOut(HttpSession session) | Одјава на корисникот |
| details(@PathVariable int productId) | Го дава погледот за детален приказ на производ со дадено ID |
| Index(Params …) | Го дава погледот за почетната страна на корисникот |
| cart(HttpSession session) | Го дава погледот за преглед на кошничката |
| checkout(HttpSession session) | Го дава погледот за checkout |
| addToCart(HttpSession session, @PathVariable int productId) | Додава производ со дадено ID во кошничката |
| removeFromCart(HttpSession session,  @PathVariable int cartProductId) | Брише производ со дадено ID од кошничката |
| UserDaoImpl | Класа која го имплементира UserDao интерфејсот. Ова е всушност објект кој го нуди поврзувањето на даден корисник со базата на податоци, односно со соодветниот корисник запишан во базата. | addUser(User u) | Додава корисник во база |
| deleteUser(int id) | Брише корисник од база |
| updateUser(User u) | Изменува тековен корисник во база |
| getUser(int id) | Враќа корисник со зададено ID |
| getAllUsers() | Враќа сите корисници |
| getUserByEmail(String email) | Враќа корисник според неговиот email |
| checkCredentials(String email, String password) | Проверува внес од корисник |
| CategoryDaoImpl | Класа која го имплементира CategoryDao интерфејсот. Ова е всушност објект кој го нуди поврзувањето на одредена категорија со соодветната категорија запишана во базата. | addCategory(Category c) | Додава категорија во база |
| deleteCategory(int id) | Брише категорија од база |
| updateCategory(Category c) | Изменува категорија од база |
| getCategory(int id) | Враќа категорија со зададено ID |
| getAllCategories() | Враќа листа од категории во база |
| getCategoryInRange(Params … ) | Враќа листа од категории во зададен опсег |
| ProductDaoImpl | Класа која го имплементира ProductDao интерфејсот. Ова е всушност објект кој го нуди поврзувањето на одреден продукт со соодветниот продукт запишан во базата. | addProduct(Product p) | Додава продукт во база |
| deleteProduct(int id) | Брише продукт од база |
| updateProduct(Product p) | Изменува одреден продукт од база |
| getProduct(int id) | Враќа продукт со зададено ID |
| getAllProducts() | Враќа листа од сите продукти |
| search(String keyword) | Пребарува продукт по клучен збор |
| getProductsInRange(int from, int to, String keyword) | Враќа листа од продукти во зададен опсег и клучен збор |
| getProductsCount(String keyword) | Враќа број на продукти пребарувани по клучен збор |
| getOnPromotion() | Враќа листа од продукти на промоција |
| CartDaoImpl | Класа која го имплементира CartDao интерфејсот. Ова е всушност објект кој го нуди поврзувањето на дадена купувачка кошничка со соодветната кошничка запишана во базата. | addCart(Cart c) | Додава кошничка во база |
| deleteCart(int id) | Брише кошничка од база |
| updateCart(Cart c) | Изменува дадена кошничка од база |
| getCart(int id) | Враќа кошничка со зададено ID |
| getAllCarts() | Враќа листа од сите кошнички |
| getByUserId(int id) | Враќа кошничка за зададен корисник |
| CartProductDaoImpl | Класа која ги поврзува Cart и Product класата, односно претставува меѓутабела која ги поврзува соодветните табели во базата на податоци. | addCartProduct(CartProduct cp) | Додава продукт во кошничка |
| deleteCartProduct(int id) | Брише продукт од кошничка |
| updateCartProduct(CartProduct cp) | Изменува продукт од кошничка |
| getAllCartProducts() | Враќа листа од сите производи во кошничка |
| getProductCount(int id) | Враќа количина на производ во кошничка |
| getCartProductsByCartId(int id) | Враќа производи во кошничка според ID |
| PurchaseDaoImpl | Класа која го имплементира PurchaseDao интерфејсот. Ова е всушност објект кој го нуди поврзувањето на дадена нарачка со соодветната нарачка запишана во базата на податоци. | addPurchase(Purchase p) | Додава нарачка во база |
| deletePurchase(int id) | Брише нарачка од база |
| updatePurchase(Purchase p) | Изменува постоечка нарачка |
| getPurchase(int id) | Враќа нарачка со зададено ID |
| getAllPurchases() | Враќа листа од сите нарачки |
| PurchaseDetailDaoImpl | Класа која го имплементира PurchaseDetailDao интерфејсот. Ова е всушност објект кој го нуди поврзувањето на деталите за дадена нарачка со соодветните детали за истата нарачка запишани во базата. | addPurchaseDetail(PurchaseDetail pd) | Додава детали за нарачка во база |
| deletePurchaseDetail(int id) | Брише детали за нарачка од база |
| updatePurchaseDetail(PurchaseDetail pd) | Изменува детали за одредена нарачка |
| getPurchaseDetail(int id) | Враќа детали за нарачка |
| getAllPurchaseDetails() | Враќа листа од сите детали за нарачки |
| PurchaseProductDaoImpl | Класа која ги поврзува Purchase и Product класата, односно претставува меѓутабела помеѓу овие две класи во базата на податоци. | addPurchaseProduct(PurchaseProduct pp) | Додава продукт во нарачка |
| deletePurchaseProduct(int id) | Брише продукт од нарачка |
| updatePurchaseProduct(PurchaseProduct pp) | Изменува продукт од нарачка |
| getPurchaseProduct(int id) | Враќа продукт од нарачка |
| getAllPurchaseProducts() | Враќа листа од продукти од нарачка |
| User | Класа која ги содржи сите атрибути за корисникот. | getters & setters |  |
| Category | Класа која ги содржи сите атрибути за дадена категорија | getters & setters |  |
| Product | Класа која ги содржи сите атрибути за продуктот | getters & setters |  |
| Cart | Класа која ги содржи сите атрибути за дадена купувачка кошничка | getters & setters |  |
| CartProduct | Класа која ги поврзува одредени продукти со дадена купувачка кошничка | getters & setters |  |
| Purchase | Класа која ги содржи атрибути за одредена нарачка | getters & setters |  |
| PurchaseDetail | Класа која ги содржи деталите за одредена нарачка | getters & setters |  |
| PurchaseProduct | Класа која поврзува одредени продукти со дадена нарачка | getters & setters |  |
| UserServiceImpl | Класа која нуди сервис, односно начин за справување со податоците за корисникот | addUser(User u) | Додава корисник и му креира кошничка |
| deleteUser(int id) | Брише корисник |
| updateUser(User u) | Изменува корисник |
| getUser(int id) | Враќа корисник со зададено ID |
| getAllUsers() | Враќа листа од сите корисници |
| signInUser(User user, HttpSession session) | Проверува дали корисник е најавен |
| addOrUpdateUser(User u) | Додава или изменува корисник |
| CategoryServiceImpl | Класа која нуди сервис за справување со податоците за категориите | addCategory(Category c) | Додава категорија |
| deleteCategory(int id) | Брише категорија |
| updateCategory(Category c) | Изменува категорија |
| getCategory(int id) | Враќа категорија со зададено ID |
| getAllCategories() | Враќа листа од сите категории |
| addOrUpdateCategory(Category c) | Додава или изменува категорија |
| ProductServiceImpl | Класа која знае како да се справи, односно нуди сервис за податоците за продуктите | addProduct(Product p) | Додава продукт |
| deleteProduct(int id) | Брише продукт |
| updateProduct(Product p) | Изменува продукт |
| getProduct(int id) | Враќа продукт со зададено ID |
| getAllProducts() | Враќа листа од сите продукти |
| search(String keyword) | Враќа листа од сите продукти по клучен збор |
| getProductsInRange(int category, int page, String keyword) | Враќа листа од продукти со зададен опсег |
| getProductPageCount(String keyword, int category) | Враќа број на продукти по страна |
| getOnPromotion() | Враќа листа од продукти на промоција |
| CartProductServiceImpl | Сервис кој се справува со податоците на продуктите во одредена купувачка кошничка | addCartProduct(CartProduct cp) | Додава продукт во кошничка |
| deleteCartProduct(int id, User user) | Брише продукт од кошничка |
| updateCartProduct(CartProduct cp) | Изменува продукт во кошничка |
| getAllCartProducts() | Враќа листа од сите продукти во кошничка |
| getProductCount(int id) | Враќа број на продукти во кошничка |
| getCartProductsFromCart(Cart cart) | Враќа листа од продукти ви кошничка со зададено ID |
| CartServiceImpl | Класа која манипулира, односно знае како да се справи со податоците за купувачките кошнички | addCart(Cart c) | Додава кошничка |
| deleteCart(int id) | Брише кошничке |
| updateCart(Cart c) | Изменува кошничка |
| getCart(int id) | Враќа кошничка со зададено ID |
| getAllCarts() | Враќа листа од сите кошнички |
| addToCart(int productId, User user) | Додавање продукт во кошничка за одреден корисник |
| PurchaseDetailServiceImpl | Класа која нуди сервис за справување со податоците за деталите на дадена нарачка | addPurchaseDetail(PurchaseDetail pd) | Додавање детали за нарачка |
| deletePurchaseDetail(int id) | Бришење детали за нарачка |
| updatePurchaseDetail(PurchaseDetail pd) | Измена на детали за нарачка |
| getPurchaseDetail(int id) | Враќа детали за нарачка |
| getAllPurchaseDetails() | Враќа листа од сите детали за нарачка |
| PurchaseProductServiceImpl | Сервис кој се справува со податоците за продуктите за конкретна нарачка | addPurchaseProduct(PurchaseProduct pp) | Додава продукт во нарачка |
| deletePurchaseProduct(int id) | Брише продукт од нарачка |
| updatePurchaseProduct(PurchaseProduct pp) | Изменува продукт од нарачка |
| getPurchaseProduct(int id) | Враќа продукт од нарачка |
| getAllPurchaseProducts() | Враќа листа од сите продукти во нарачка |
| PurchaseServiceImpl | Класа која нуди сервис, односно начин за справување на податоците за нарачките | addPurchase(Purchase p) | Додава нарачка |
| deletePurchase(int id) | Брише нарачка |
| updatePurchase(Purchase p) | Изменува нарачка |
| getPurchase(int id) | Враќа нарачка |
| getAllPurchases() | Враќа листа од сите нарачки |
| AdminModelAndView | Класа која содржи само конструктори со различни параметри за администраторскиот дел | constructors |  |
| CustomerModelAndView | Класа која содржи конструктори со различни параметри за корисничкиот дел | constructors |  |

* 1. Опис на базата

Базата се состои од следните табели:

|  |
| --- |
| PurchaseDetail |
| -id: int(11), primary key |
| -productName: varchar(255) |
| -quantity: int(11), number |
| -unitCost: double, number |
| -product\_id: int(11), foreign key, number |

|  |
| --- |
| Category |
| -id : bigint(20), primary key |
| -description: varchar(255) |
| -name: varchar(255) |

|  |
| --- |
| Purchase |
| -id: int(11), primary key |
| -canceled: tinyint(1), number |
| -comments: varchar(255) |
| -completed: tinyint(1), number |
| -dateCreated: datetime |
| -dateShipped: datetime |
| -shippingAddress: varchar(255) |
| -verified: tinyint(1), number |
| -purchaseDetail\_id: int(11), foreign key, number |
| -purchaseProduct\_id: int(11), foreign key, number |
| -user\_id: int(11), foreign key, number |

|  |
| --- |
| Product |
| -id: int(11), primary key |
| -description: varchar(255) |
| -largeImage: varchar(255) |
| -smallImage: varchar(255) |
| -name: varchar(255) |
| -onPromotion: tinyint(1), number |
| -price: double, number |
| -fileLargeImage: tinyblob |
| -fileSmallImage: tinyblob |

|  |
| --- |
| Cart |
| -id: int(11), primary key |
| -user\_id: int(11), foreign key, number |

|  |
| --- |
| User |
| -id: int(11), primary key |
| -address: varchar(255) |
| -authority: varchar(255) |
| -email: varchar(255) |
| -name: varchar(255) |
| -surname: varchar(255) |
| -password: varchar(255) |
| -enabled: int(11), number |

|  |
| --- |
| VerificationUser |
| -id: int(11), primary key |
| -link: varchar(255) |
| -user\_id: int(11), foreign key |

И меѓутабелите:

|  |
| --- |
| product\_category |
| -products\_id: int(11), number, primary key |
| -categories\_id: bigint(20), number, primary key |

|  |
| --- |
| cartproduct |
| -id: int(11), primary key |
| -quantity: int(11), number |
| -cart\_id: int(11), foreign key, number |
| -product\_id: int(11), foreign key, number |

|  |
| --- |
| purchaseproduct |
| -id: int(11), primary key |
| -quantity: int(11), number |
| -product\_id: int(11), foreign key, number |

|  |
| --- |
| purchase\_purchasedetail |
| -purchase\_id: int(11), foreign key |
| -purchaseDetails\_id: int(11), foreign key |

Помеѓу класите *Product* и *Category* постои релација n:m овозможена преку меѓутабелата *product\_category*.

Исто така, помеѓу *Purchase* и *Product* постои n:m релација преку меѓутабелата *purchaseproduct*.

Product и PurchaseDetail се поврзани со 1:n релација.

*Purchase* и *PurchaseDetail* се поврзани со n:m релација преку purchase\_purchasedetail.

Класите *Product* и *Cart* се поврзани со n:m релација, овозможена преку меѓутабелата *cartproduct*, бидејќи во една купувачка кошничка може да има повеќе производи, но и производите можат да бидат во повеќе купувачки кошнички.

Класите *User* и *Cart* се поврзани со 1:1 релација, бидејќи даден корисник може да има само една купувачка кошничка.

Исто и класите *Purchase* и *User* имаат 1:n релација поради тоа што еден корисник може да прави повеќе нарачки.

Класите User и VerificationUser имаат 1:1 релација - еден корисник има еден линк за верификација.

