CS341 Shopping Cart Functional Requirements

Group 1: Sal Skare, Jack Englund, David Wissink, and John Collins

November 29, 2018

1 About this document

This document describes the functional requirements for the software for an online e-commerce shopping cart software product.

2 Functional Requirements

2.1 Functional Requirements For Cart

Index:	CT.1
Name:	GetCartPrice
Purpose:	calculates the total price, taking ship and promotion codes into consideration
Input parameters:	Email, PromotionCode
Action:	fetches products from the cart, applies promotion to item if valid, then adds price to running total
Output parameters:	Total price
Exceptions:	invalid promotion, empty cart
Remarks:	None.
Cross references:	None.

Index:	CT.2
Name:	AddItemToCart
Purpose:	Adds the item to the user's cart.
Input parameters:	Access Token, Item ID, quantity
Action:	Adds the number of items given by the quantity to the authenticated user's cart.
Output parameters:	A success or failure message.
Exceptions:	The user is not authenticated, the item does not exist, there was a database error.
Remarks:	None.
Cross references:	None.

Index:	CT.3
Name:	RemoveItemFromCart
Purpose:	Removes the item from the user's cart.
Input parameters:	Access Token, Item ID
Action:	Completely removes a product from the authenticated user's cart.
Output parameters:	A success or failure message.
Exceptions:	The user is not authenticated, the item does not exist, there was a database error.
Remarks:	None.
Cross references:	None.

Index:	CT.4
Name:	EmptyCart
Purpose:	Removes all items from a car
Input parameters:	Access Token
Action:	removes all items from the authenticated user's cart.
Output parameters:	None.
Exceptions:	The user is not authenticated, there was a database error.
Remarks:	takes back to the "home" page when complete
Cross references:	None.

Index:	CT.5
Name:	GetCart
Purpose:	Returns a list of all items in a user's cart.
Input parameters:	Access Token
Action:	Creates a list of item IDs and quantity in an authenticated user's cart.
Output parameters:	A list of Item IDs and the count for each item.
Exceptions:	The user is not authenticated, there was a database error.
Remarks:	None.
Cross references:	None.

Index:	CT.6
Name:	UpdateCart
Purpose:	Updates the quantities of items in the cart
Input parameters:	Access Token, items, corresponding quantities
Action:	Updates all of the item quantities in the cart and recalculates the price.
Output parameters:	A success or failure message.
Exceptions:	The user is not authenticated, there was a database error, user supplied invalid quantity
Remarks:	None.
Cross references:	None.

2.2 Functional Requirements For Categories

Index:	CG.1
Name:	CreateCategory
Purpose:	to create a new product category
Input parameters:	Name, admin user's token
Action:	authenticates admin privileges and then creates a new product category
Output parameters:	A success or failure message.
Exceptions:	invalid token, database error
Remarks:	None.
Cross references:	None.

Index:	CG.2
Name:	DeleteCategory
Purpose:	removes an existing category
Input parameters:	Name, Admin user's token
Action:	authenticates the admin's token and then deletes the category
Output parameters:	A success or failure message.
Exceptions:	invalid token, database error
Remarks:	None.
Cross references:	None.

2.3 Functional Requirements for Registered User Checkout

Index:	CO.1
Name:	CompleteCheckout
Purpose:	complete the paypal checkout process
Input parameters: Registered user token, Promotion Code	
Action:	Calculates total for a registered user's cart and sends request to paypal API
Output parameters:	Success or failure message.
Exceptions:	Cart is empty, token invalid, paypal API failure
Remarks:	None
Cross references:	CT.1

Index:	CO.2
Name:	PayPalEdit
Purpose:	Edit's the site's paypal details
Input parameters: Registered user's token, PayPal details Action:	Authenticates user and alters the details our site used to make requests to Paypal's API
Output parameters:	Success or failure message.
Exceptions:	invalid token, Paypal API failure
Remarks:	
Cross references:	None.

2.4 Functional Requirements For Guest Cart

Index:	GC.1
Name:	AddItemToGuestCart
Purpose:	Adds the item to the guest user's cart.
Input parameters:	Guest user's id (A randomly generated GUID), Item ID, item quantity.
Action:	Adds the number of items given by the quantity to the specified guest user's cart. If this is the first time calling this function, an entry for the guest user will first be created in the database.
Output parameters:	A success or failure message.
Exceptions:	The item does not exist, there was a database error.
Remarks:	The front-end should store the guest user ID in a browser cookie so that subsequent requests can be made.
Cross references:	None.

Index:	GC.2
Name:	RemoveItemFromGuestCart
Purpose:	Removes the item from the guest user's cart.
Input parameters:	Guest user's id, Item ID
Action:	Removes the all of a specific item from the specified guest user's cart.
Output parameters:	A success or failure message.
Exceptions:	The guest does not exist, the item does not exist, there was a database error.
Remarks:	None.
Cross references:	None.

Index:	GC.3
Name:	EmptyGuestCart
Purpose:	Removes all items from the guest user's cart.
Input parameters:	Guest user's id
Action:	Removes all items from the specified guest user's cart.
Output parameters:	A success or failure message.
Exceptions:	The guest does not exist, the item does not exist, there was a database error.
Remarks:	None.
Cross references:	None.

Index:	GC.4
Name:	GetGuestCart
Purpose:	Returns a list of all items in a guest user's cart.
Input parameters:	Guest user's ID
Action:	Creates a list of item IDs and count in a guest user's cart.
Output parameters:	A list of Item IDs and the count for each item.
Exceptions:	The guest user does not exist, there was a database error.
Remarks:	None *Cross references:*

None.

Index:	GC.5
Name:	UpdateGuestCart
Purpose:	Updates the quantities of items in a guest user's cart.
Input parameters:	Guest user's ID, item IDs, corresponding quantities
Action:	Updates the quantities of items in a guest user's cart and recalculates the price
Output parameters:	None.
Exceptions:	The guest user does not exist, there was a database error.
Remarks:	GC.6 . *Cross references:*
None.	

Index:	GC.6
Name:	GetGuestCartPrice
Purpose:	calculates the total price, taking ship and promotion codes into consideration
Input parameters:	Guest user's id, PromotionCode
Action:	fetches products from the cart, applies promotion to item if valid, then adds price to running total
Output parameters:	Total price
Exceptions:	invalid promotion, empty cart
Remarks:	None.
Cross references:	None.

2.5 Functional Requirements for Guest Checkout

Index:	GO.1
Name:	Complete
Purpose:	Completes a payment, sending it off to paypal
Input parameters:	Guest user's id, Promotion code
Action:	Calculate cart total and makes request to Paypal API
Output parameters:	None.
Exceptions:	invalid Guest user id, Paypal API failure
Remarks:	None.
Cross references:	None.

2.6 Functional Requirements For Registered User

Index:	U.1
Name:	CreateRegisteredUser
Purpose:	To create a new user in the database.
Input parameters:	Email, Password, guest ID, type(admin, customer)
Action:	Creates a new entry in the database for the Registered User, hashing the password. If the guest user ID is given, the guest user cart will be imported to the new user.
Output parameters:	Success or failure message.
Exceptions:	The user already exists, not all required fields were given, there was a database error, the user does not have permission to create a new admin.
Remarks:	It will be up to the front-end design to also log in the newly created user if that functionality is desired. Also, to create an admin user requires an existing admin to make the request, including a valid Access Token
Cross references:	None.

Index:	U.2
Name:	AuthenticateUser
Purpose:	Authenticates the user to the server for a set period of time.
Input parameters:	Email, Password
Action:	Generates an Access Token for the given user.
Output parameters:	A temporary Access Token to be included in all requests that require elevated permission.
Exceptions:	The user does not exist, the password did not match the user, there was a database error.
Remarks:	The server will log the time the token was created and mark it as expired after a set length of time, requiring the user to log in again.
Cross references:	None.

Index:	U.3
Name:	GetUser
Purpose:	Returns information about the user
Input parameters:	Email, Access Token
Action:	Queries the database for the user, then returns all available fields.
Output parameters:	An *User object, containing, email, name, type of user.*
Exceptions:	The user is not authenticated, there was a database error.
Remarks:	None.
Cross references:	None.

2.7 Promotions

Index:	P.1
Name:	CreatePromotion
Purpose:	To create a promotion for users to apply to items
Input parameters:	Name, Discount Code, Discount Type *[bogo, percent], Percent, End date*
Action:	Creates a new promotion in the database.
Output parameters:	Success or Failure message.
Exceptions:	The discount code already exists, invalid end date
Remarks:	The promotion should only be valid until the end date, the code can only be used once per user.
Cross references:	None.

Index:	P.2
Name:	EndPromotion
Purpose:	Forcibly ends a promotion.
Input parameters:	Discount Code *[Identifier].*
Action:	Sets the end time of the promotion in the database to the current time.
Output parameters:	Success or Failure message.
Exceptions:	The discount code doesn't exist.
Remarks:	It doesn't matter if they want to end a promotion that's over since it's already over.
Cross references:	None.

Index:	P.2
Name:	EditPromotion
Purpose:	Edits an existing promotion, only changing fields given, all based on the promotion code.
Input parameters:	Discount Code and any fields to change *[Identifier].*
Action:	Edits the given promotion in the database.
Output parameters:	Success or Failure message.
Exceptions:	The discount code doesn't exist, the promotion is already ended.
Remarks:	None.
Cross references:	None.

Index:	P.3
Name:	GetPromotion
Purpose:	Gets data for one promotion.
Input parameters:	Discount Code *[Identifier].*
Action:	Gets a promotion based on the discount code given.
Output parameters:	JSON string of the promotion's data.
Exceptions:	The discount code doesn't exist, the promotion is over.
Remarks:	None.
Cross references:	None.

Index:	P.3
Name:	GetAll
Purpose:	Get all existing promotions and their data.
Input parameters:	Discount Code *[Identifier].*
Action:	Gets all promotions from the database.
Output parameters:	JSON string of all the promotion data.
Exceptions:	None.
Remarks:	None.
Cross references:	None.

2.8 Product Page

Index:	PP.1
Name:	GetProduct
Purpose:	Gets info on an existing Product
Input parameters:	Product Id
Action:	Returns the product name, price, quantity, image, and description.
Output parameters:	JSON string of the item's data
Exceptions:	Invalid ID.
Remarks:	None.
Cross references:	None.

Index:	PP.1
Name:	GetAllProduct
Purpose:	Gets info about a group of products.
Input parameters:	Category
string to search for.	
Action:	Find all items that meet the given parameters
Output parameters:	JSON string of the item's data
Exceptions:	None.
Remarks:	None.
Cross references:	None.

Index:	PP.2
Name:	NewProduct
Purpose:	Creates a new product entry in the db.
Input parameters:	Name, Category, Price, Seller, Initial Quantity, Max Quantity, Reviews, Images
Action:	Creates a new entry in the product database.
Output parameters:	Success or Failure message.
Exceptions:	Max Quantity is less than 1, Image files are too big.
Remarks:	None.
Cross references:	None.

Index:	PP.3
Name:	EditProduct
Purpose:	Edits an already existing product
Input parameters:	PID, Name, Category, Price, Seller, Initial Quantity, Max Quantity, Reviews, Images
Action:	Edits the row of PID in the database.
Output parameters:	Success or Failure message.
Exceptions:	PID is not valid, then same exceptions as NewProduct
Remarks:	None.
Cross references:	None.

Index:	PP.4
Name:	DeleteProduct
Purpose:	Removes a product from the database
Input parameters:	PID
Action:	Removes the row of the given PID from the database.
Output parameters:	Success or Failure message.
Exceptions:	PID does not exist.
Remarks:	None.
Cross references:	None.

2.9 Functional Requirements for Login

3 Assumptions:

1. Users are identified by their email addresses.
2. If a user wants to change their email address, they will need to create a new account.
3. All payment info and address data will be handled through PayPal.
4. only one promotion code can be applied to the cart.
5. Admin are also registered customers who may purchase items.
6. Admin are distinguished by their ability to access the admin pages

7. A category is not deleted unless all items of that category have also been deleted