SEMESTRAL PROJECT – TS1 (by Roman Danilchenko and Volodymyr Semenyug)

1. Test plan

1.1 Application description

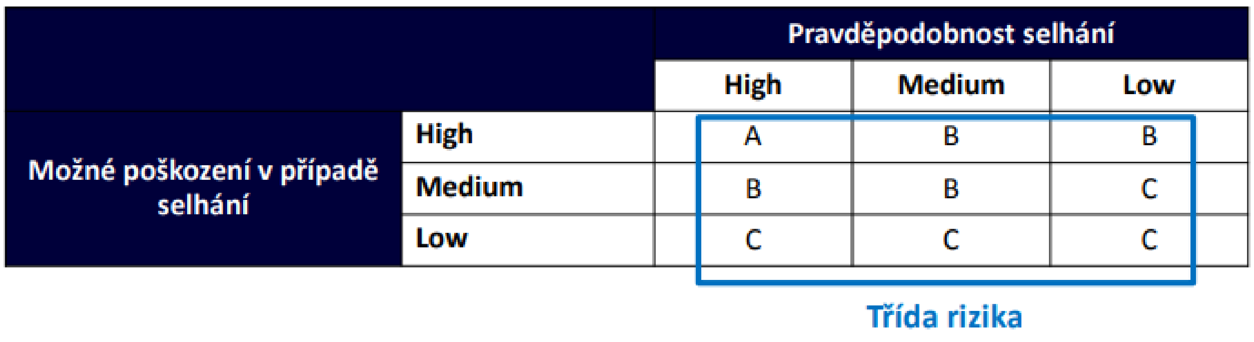
Application opencart.com can be opened in two modes: user and administrator.
For our semestral project we have chosen user mode to be tested.
As user you can login in\ sign in, operate with products (add them to shopping cart, wish list, buy them etc.), communicate with tech. support, return ordered products.

1.2 Application parts

a) Shopping : find item, add item to cart\wish list, checkout;
b) Checkout: log in\ sign in, enter billing and payment information
c) Contact tech support
d) Return order

1.3 Prioritize application parts

Priorities:



		Pravděpodobnost selhání		
		High	Medium	Low
Možné poškození v případě selhání	High	A	B	B
	Medium	B	B	C
	Low	C	C	C

Třída rizika

Application part	Probability of failure	Damage	Risk class
Shopping	H	H	A
Checkout	H	H	A
Tech. support	M	L	C
Return item	M	H	B

1.4 Test levels

Part	Risk class	Revision	Unit tests	System tests	UAT	Test in production
Shopping	A		M	H	H	L
Checkout	A		M	H	H	L
Tech. support	B		M	H	H	L
Return item	B		M	H	H	L

2. Test scenario

2.1 Input tests

a) Product returns

Name, Surname

EC Type	Example
Invalid EC (technical point of view)	Sequence of characters with the length greater than 32 or less than 1, leave it free
Invalid EC (business point of view)	Non-existing name and surname (sequence of random characters)
Valid EC	Real name and surname

Email

EC Type	Example
Invalid EC (technical point of view)	String that does not have pattern _+@_+._+ , leave it free
Invalid EC (business point of view)	Non-existing email
Valid EC	Email with pattern _+@_+._+

Phone

EC Type	Example
Invalid EC (technical point of view)	Sequence of characters with the length greater than 32 or less than 3, leave it free
Invalid EC (business point of view)	Random string of alphanumeric characters
Valid EC	Real phone number

Order ID

EC Type	Example
Invalid EC (technical point of view)	leave it free
Invalid EC (business point of view)	Non-existing order id, random string of alphanumeric characters

Valid EC	Existing order id

Product name

EC Type	Example
Invalid EC (technical point of view)	Sequence of characters with the length greater than 255 or less than 3, leave it free
Invalid EC (business point of view)	Non-existing product name, random string of alphanumeric characters
Valid EC	Existing product name

Product code

EC Type	Example
Invalid EC (technical point of view)	Sequence of characters with the length greater than 255 or less than 3, leave it free
Invalid EC (business point of view)	Non-existing product code, random string of alphanumeric characters
Valid EC	Existing product code

Reason for return

EC Type	Example
Invalid EC (technical point of view)	Leave it free
Invalid EC (business point of view)	None
Valid EC	Any option

Product is opened

EC Type	Example
Invalid EC (technical point of view)	Leave it free
Invalid EC (business point of view)	None
Valid EC	Any option

b) Gift certificate

Recipient's name and your name

EC type	Example
Invalid EC (technical point of view)	Too long strings.
Invalid EC (business point of view)	Name must be 1 to 64 characters long, null.
Valid EC	Rest, e.g. Roman and Volodymyr

Recipient’s e-mail and your e-mail

EC type	Example
Invalid EC (technical point of view)	Too long strings or objects of wrong type.
Invalid EC (business point of view)	Wrong e-mail addressess (e.g. address doesn’t have an at sign or the domen doesn’t exist) or null.
Valid EC	Rest, e.g. first@gmail.com and second@gmail.com

Gift certificate theme

EC type	Example
Invalid EC (technical point of view)	None
Invalid EC (business point of view)	Wrong checkbox
Valid EC	Birthday, Christmas, General

Message

EC type	Example
Invalid EC (technical point of view)	None
Invalid EC (business point of view)	None
Valid EC	Everything

Amount

EC type	Example
Invalid EC (technical point of view)	Object of wrong type
Invalid EC (business point of view)	Value must be between $1 and $1.000
Valid EC	Every integer in interval [1, 1000], e.g. 50

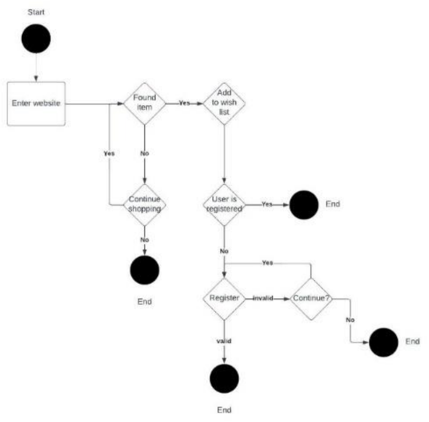
2.2 Pairwise testing

All input combination for pairwise tests can be found in files ProductReturns.xls and GiftCertificate.xls

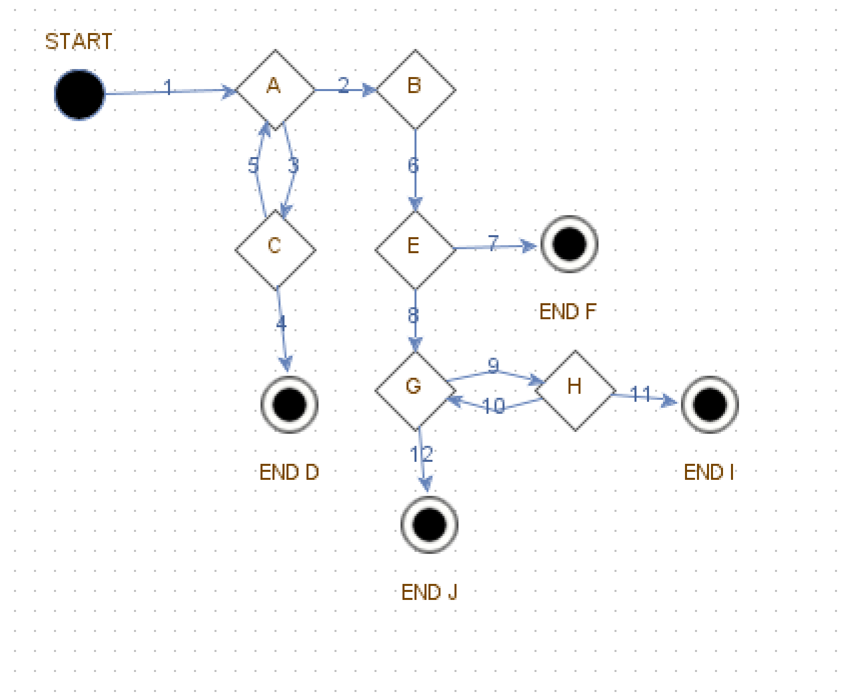
2.3 Process tests

Processes:

1) **Add item to wishlist**



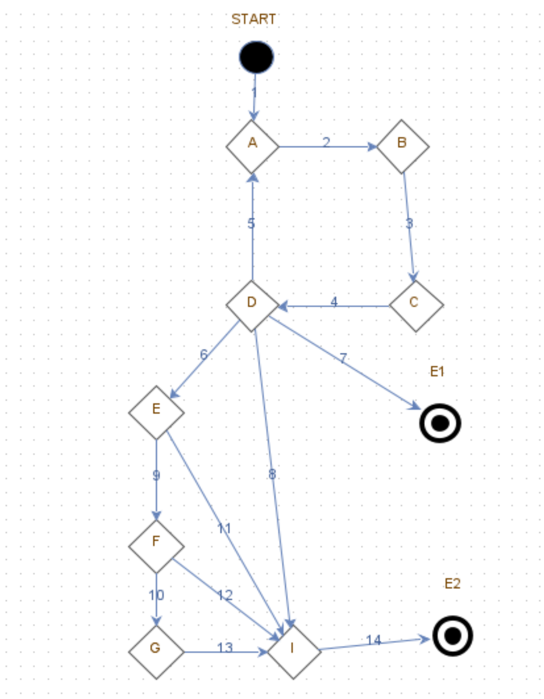
Simplified diagram:



Test cases (TDL 2):

No.	Test sequence
1	1 - 3 - 4
2	1 - 3 - 5 - 3 - 4
3	1 - 2 - 6 - 7
4	1 - 3 - 5 - 2 - 6 - 7
5	1 - 2 - 6 - 8 - 9 - 10 - 9 - 11
6	1 - 2 - 6 - 8 - 12
7	1 - 2 - 6 - 8 - 9 - 10 - 12

2) **Add item to shopping art and try to continue to checkout:**



Where:

A – Enter user main page

B – Choose item

C – Add item to cart

D – Enter shopping cart (after that you can either continue shopping (5) or try to buy item (6,8,7))

E1 – if item is not in stock, it is impossible to buy it

E – Use coupon to get discount

F – Count estimate shipping

G – Use gift certificate

I – Go to checkout

E2 – if you successfully entered checkout page (you can buy Item, test is over)

2.4 Test scenarios in details

Test ID	buyAsGuest
Name	Test of placing orders as guest
Details depth	Medium
Test summary	Test ability of a customer to buy item as guest
Description	We found an item, that we want to buy. In checkout section we choose checkbox „Buy as guest“, after that we need to fill all the information about us. In the end we need to press „Continue“ button, and our order will be to be placed.
Inputs conditions	Item must be in stock.
Testing data	Text area „Your order has been placed“
Expected output	We can verify our output by checking, if a certain element is present on the web site, and than use assertTrue function.

Test ID	addItemToShoppingCartUseCouponCountEstimateShippingUseCertificateGoToCheckout_ItemIsAvailable
Name	Shopping cart options testing
Details depth	High
Test summary	Use all options in shopping cart
Description	Find item, add it to shopping cart, proceed to shopping cart, try to use coupon, count estimated shipping, use fist certificate, proceed to checkout
Inputs conditions	Item must be in stock
Testing data	Text area “Checkout”
Expected output	AssertTrue function is successful

3. Test implementations

Test implementations can be found here: https://github.com/big-boy300/TS1_sem