D0018E Report

Lorentz Kinde, 940703-6350*
Anton Tiberg, 960725-4654†
Luleå Tekniska Universitet, 971 87 Luleå, Sverige

December 13, 2017

*lorkin-4@student.ltu.se
†anttib-5@student.ltu.se

1 Project background

In this project a e-commerce website for a office supply company will be developed. The webpages front end will be written in HTML+CSS+php and the back end in php and linked with a SQL database.
This report will showcase several tools used to design the webpage as well as give insight to how it's built and implemented, primarily focused around a scrum based development system [1].
We would've used Github to share files, but ended up using google drive to have a joint version control repository system out of simplicity. Link to the finnished github can be found in the references [3]. Finished webpage can also be found in references [2].

2 Functionality

2.1 User stories

- As a user I want to be able to add products to a shopping cart.
- As a user I want to be able to write a review for the products I have ordered.
- As a user I want to be able to login to the website and have shopping cart saved for a later session.
- As a user I want to be able to change my account information.
- As a user I want to be able to view my order history.
- As an admin I want to be able to add products to the product lists.
- As an admin I want to be able to remove unnecessary reviews.
- As an admin I want to be able to fill up the product stock.

2.2 User cases

User case: 1

Actor	System user
Precondition	The shopping cart is empty.
Postcondition	The shopping cart contains the correct quantity of the selected products.
Main path (M)	1. User enters website 2. User navigates using categories to wanted product 3. User clicks product "add to cart" button 4. Product item is added to user virtual shoppning cart

User case: 2

Actor	System user
Precondition	A product has been ordered.
Postcondition	A review of ordered product is posted on the website
Main path (M)	1. User navigates to product webpage 2. User writes a comment about said product 3. User clicks "submit review" 4. Review is then added to database and shown on the product webpage

User case: 3

Actor	System user
Precondition	A user has created an account on the webpage and has ordered some products
Postcondition	Webpage displays users previously made orders
Main path (M)	1. User enters and logs in to the webpage 2. User navigates to account overview 3. Webpage displays along other things all previously made orders

User case: 4

Actor	System user
Precondition	The user has administratory rights on his account
Postcondition	A new product has been added to the product list
Main path (M)	1. Admin navigates to product page 2. Admin navigates to the product 3. Admin enters required information and clicks 'post product' 4. Product becomes listed on the webpage

User case: 5

Actor	System user
Precondition	Trolls exists online and write unethically mean comments on reviews
Postcondition	The troll reign ends once and for all as the review is permanently destroyed
Main path (M)	1. Admin navigates to product page where unethical comments are being posted 2. Admin finds targeted review 3. Admin clicks 'Remove review'and gets promted with a confirmation dialog 4. The review is banished from the product page and database and all becomes well in the world

2.3 Use cases

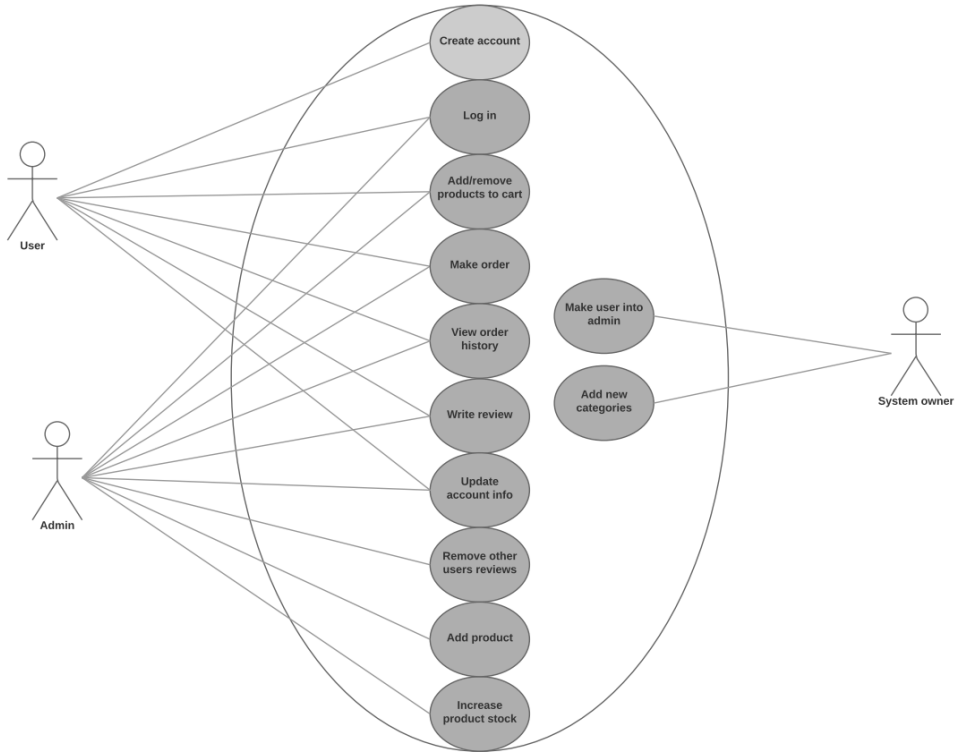

Figure 1: Use cases of the three roles on the webpage

2.4 Requirements

- The user must have a basic grasp at webpage navigation
- The user must remember his own password and account name (no password reset function will exist at first version of webpage)
- The user must have money to purchase our products

2.5 Assumptions

- We assume that there is a existing money transfer service (Paypal, Swish, Klarna)
- We assume that there will be a existing delivery service (Postnord, DHL, Schenker)

- We assume that there will exist a admin account on the webspage on delivery to the client

3 Method and tools

MySQL Workbench was used for creating the database schema and setting up the database. For developing the front- and back end web-site ordinary text editors (Kylie, N++, Vim) was used to code in HTML, CSS, PHP and SQL.

4 Backlog

Sprint 3

Objectives

Front end: Small fixes.
Back end: Setup the review system such that users can add a review to ordered products.

What needs to be done (listed after priority):

1. Setup review system (High prio, Total time estimated: 2hrs)
2. Setup change account information (High prio, Total time estimated: 2hrs)
3. Setup stock management (High prio, Total time estimated: 1hrs)
4. Fix so admin can add pictures when adding product (Low prio, Total time estimated: 1hrs)
5. Setup password encryption (Low prio, total time estimated: 1hrs)

Demo:
Login as admin, add more products to stock, add/remove reviews, change account information.
Login as user, add/remove reviews, change account information.

Figure 2: The point list version of the project backlog

SPRINT 1:				
	To do	**Under Progress**	**Demo**	**Completed**
				Basic website
				Develop PHP
				Database Schedule
				Database Setup
				Plan sprint 2
SPRINT 2:				
	To do	**Under Progress**	**Demo**	**Completed**
				Setup user login
			Setup cart system	
			Setup order system	
			Admin product management	
			Setup admin product page	
				Setup product pages
SPRINT 3:				
	To do	**Under Progress**	**Demo**	**Completed**
			Review: Add review	
			Review: Remove review	
			Review: View review	
				Database management security
		Debugging and testing		
				Fix stock management
				Fix product pictures (however)
				Change account information functionality
		Fix details	Fix details	

Figure 3: The table version of the project backlog

5 Database schema

The general design decision we have is to keep the amount of keys low, and keep the connectivity between to a minimum, to not connect things where it does not need to be connected.

Most values are typed as not null to ensure no empty fields, as well as most html forms being required depending on their uses.

5.1 User

Every user must have a unique email since the email is used as login identifier, but it's not primary key since that would either force us to have partial keys as foreign keys or have two identifiers which signify the same thing. We could've chosen to not use id as primary key, but we liked the idea of every user having a unique id to lower the amount of variables used in the website and keep ethical principles by protecting user emails so they're not being used by the system more than necessary.

5.2 Review

For each product the user may leave one review. The user can later alter the review by typing into the same field and pressing the same button.

5.3 Orders

In order to use the same order number when ordering several different products we use two tables. When an order is put we add the user id to a autoincremented id in orders. This order id is later used by the table ProdInOrder.

5.4 ProdInOrder

ProdInOrder copies the information from the ShoppingCarts to keep the saved informations consistency from the time when the user presses the "Add to cart" until the order has been completed and viewed in the account settings.

5.5 ShoppingCarts

Shopping carts does not automatically retrieve the price from the products but takes the price at the time the order customer adds it to the cart. This also means that if you added 3 of a product for 50, the total price will be 150 in the cart, even if the product price increases to 100 on the rest of the site. But if the user later goes and tries to order a another of the same product, where he/she notices the updated price, all of the objects in the cart will be updated.

5.6 Products

Every newly added product gets an auto incremented ID, much like the other tables, and it gets a previously set category which later lists it on the product page. The image is bound to the category, which is automatically added in php.

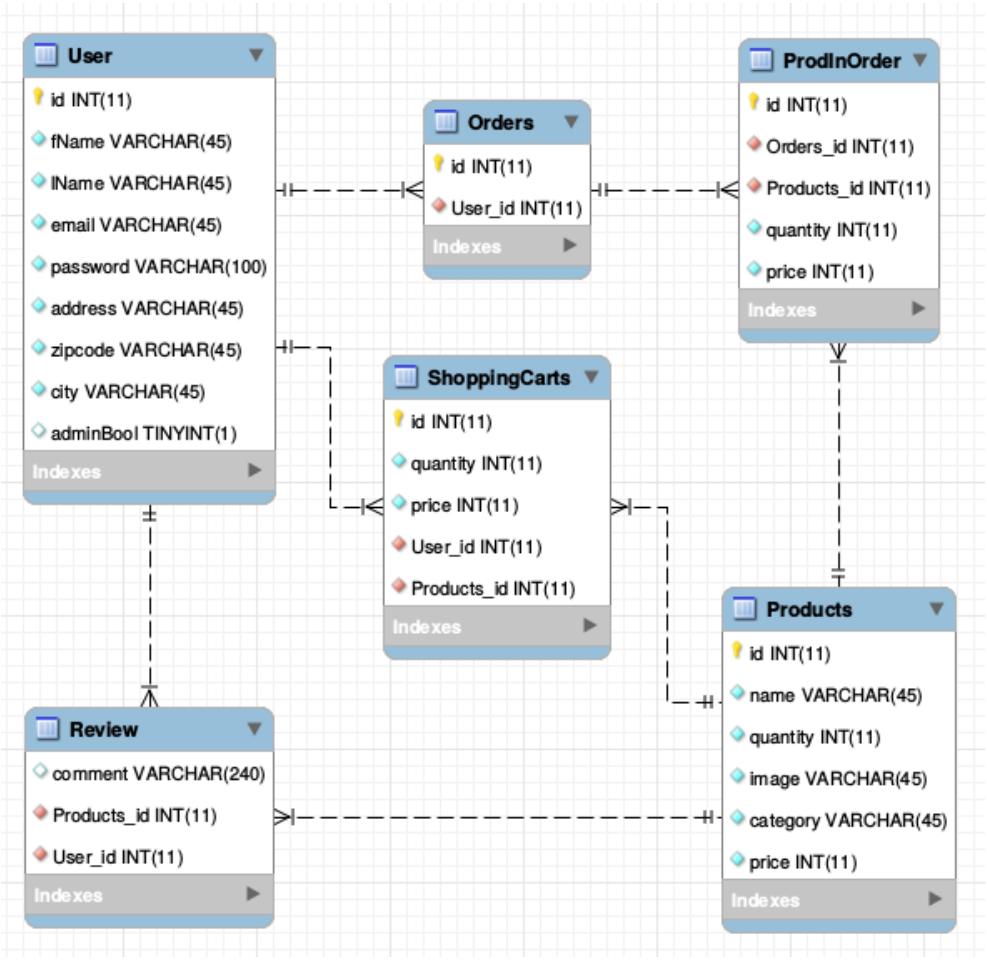

Figure 4: The database schema

6 Test case specifications

Regular user:

1. Create user account.
2. Log in.
3. Browse the products.
4. Add Products to cart.
5. Remove unwanted products from the cart.
6. Order the products.
7. View order history.
8. Write a review.
9. Edit your review if wanted.
10. Change account information.
11. Log out.

Admin:

1. Log in to admin account.
2. Add wanted products to the list.
3. Increase stock quantity for different products.
4. Remove unnecessary reviews.
5. Watch orders.
6. Add products to cart.
7. Remove unwanted products from the cart.
8. Order the products.
9. View order history.
10. Write a review.
11. Remove your review if wanted.
12. Change account information.
13. Log out.

7 Limitations and improvements

Admin cannot add a new picture of a product to the webpage since we dont have complete ownership on ther webpage, it's hosted by ltu. Instead we manually added a representative picture per category, but this could easily be fixed if the website is uploaded on a webhotel or similar.

The frontend could also use a proper update, we've mostly focused on functionality and back end since that's the requirement of this course. The review system could be extended to include a rating 1-5, diverse forum like 'upvotes' and such could be implemented to the reviews.

We could've added all the function files to one php file named functions.php and then filtered which function to use depending on the $_POST value, but we realized this far too late and changing this would cause high amounts of possible regression.

8 Security

Our site could be targeted by a SQL injection in the register site where it's reformatting string escape signs, but since that's not a focus on the site and the site basically just doesn't add the user. Since the only place SQL injections could work is at insert commands and the webpage instantly redirects, injections becomes hard to preform. A possible use would be to insert a illigal username, but that is fairy secure also since once the user has been logged in the id of the user name is used on the rest of the site (and also saved in the session).

Passwords are hashed with sha256 encryption.

9 Links and references

References

[1] SCRUM, as described by scrumalliance.
`https://www.scrumalliance.org/why-scrum`

[2] Finished webpage, hosted on 's servers
`http://utbweb.its.ltu.se/~anttib-5/`

[3] Link to Github repository with the finished php files and images.
`https://github.com/iamimago/D0018E_office_webpage`