Ecommerce System Assignment

Le Barbz

Team Members

Thomas O'Gara 18379576

Jarrett Pierse 18375813

Katarina Cvetkovic 18347921

Adam Leacy 18313471

Video Walkthrough:

https://www.youtube.com/watch?v=j4DiKg1tJEU

Requirements Review

General Requirements Progress

Req. #	Title	Comment	Status
G0	View the Products Currently on offer	Products can be viewed by category.	COMPLETE
G1	View details about a particular product	Click on a product to view its details. Allows users to add products to the cart and wishlist.	COMPLETE
G2	Add a product to their shopping cart	Integrated with HTTP Sessions - See Additional Requirements for more information	COMPLETE
G3	Remove a product from their shopping cart	Integrated with HTTP Sessions - See Additional Requirements for more information	COMPLETE
G4	Search for a product.	Search bar is present in the header	COMPLETE
G5	Create a customer account	Integrated with Spring Security - See Additional Requirements for more information	COMPLETE
G6	Log in to their account.	Integrated with Spring Security - See Additional Requirements for more information	COMPLETE

Logged in Customer Requirements Progress

Req. #	Title	Comment	Status
C0	Go to the checkout to purchase orders in their shopping cart.	After adding products to cart and logging in, you can go to checkout through the cart page. Once order is made, it is placed into the users order history	COMPLETE
C1	Use a (fake) payment portal to pay for their goods.	Payment portal is present on the checkout page at the bottom	COMPLETE
C2	View their order history	Go to a user's account page and the "View orders" to see a customers order history	COMPLETE

Logged in Owner Requirements Progress

Req. #	Title	Comment	Status
O0	Login via the same interface as (G7)	Integrated with Spring Security - See Additional Requirements for more information	COMPLETE
O1	Add products to the shop	Performed on the admin dashboard	COMPLETE
O2	Hide existing products (that are no longer available)	Admins can view product inventory through the dashboard and hide a product	COMPLETE
O3	View all orders	Admin dashboard allows for sorting viewing orders	COMPLETE
O4	Change the state of orders	Admin can change order state by viewing the details of a order	COMPLETE
O5	Edit their product details.	All product details can be changed (including changing the image)	COMPLETE

Technologies Used

To implement user and admin sign-up/login, we opted to integrate our site with Spring Security. Spring Security is a popular, secure, robust framework for handling authentication and authorization of Spring applications. We used password encoding in tandem with Spring Security to ensure the safety of user data. We used a variety of user roles to differentiate between types of users, such as customers and admins, so that specific routes on the website would be secure. One source route is the /admin route - users must be logged in as an admin to view any pages under this route.

When adding products/updating existing products we needed to implement a functionality to allow for new images to be uploaded. This required us to add a new controller which will handle file upload on the server side. Admins are allowed to upload images of type png and jpg via a HTML form, which is then posted to the saveProduct mapping. This method handles the file upload by saving the image in the appropriate location, for later use. The File Upload utility enables this.

To allow for admins to have a seamless experience running the site, and to ensure that they need to rely on as few external tools as possible to efficiently manage the platform, we implemented an EMail Service for the website. This EMail Service is integrated with the Spring Mail package, and uses GMail as an email server. Shop Administrators can log into the pre-configured account to view any received emails as well as send email to customers. There are various pages on the sites which integrate with this service - for example the contact form on the About Us page, as well as the Send Email button on the order view page for administrators. The sign-in credentials for the account can be found in the README.

In order to enable multiple concurrent guest users of the site, we opted to integrate with the HTTP Session technology. This is a feature of all modern web browsers, which allows us to track the cart of many guest users concurrently. The exact details of how HTTP Sessions work is slightly beyond the scope of this document, however it is enough to point out that they are supported on all modern web browsers - even as far back as IE11. This integration works well with Spring Security, as it will automatically resume and invalidate sessions on login/logout.

To fulfill our database requirements, we decided to use a docker container based on the mysql image available from Oracle. We decided to use docker as we felt it was a robust, stable, and portable solution to our database needs. Docker can be run on Windows, Mac, and Linux. Docker also eliminates the possibility of version mismatches of the database software between team members, which can cause issues during development.

Additional Requirements

For our website, we implemented several additional features of our own choice. Firstly, we added the ability for a user to create a wishlist and be able to add/remove products from their wishlist. In addition we also implemented a method in which if a guest user created a wishlist and/or a cart, then the next user to sign in will have the contents of the guests wishlist/cart merged into the users wishlist/cart. This is to prevent a customer from potentially losing any products they added while not logged in.

We provide product recommendations for other similar products depending on what a user is viewing. For instance, if a user was viewing a t-shirt then below the products details we would show the user several t-shirts that are also available from our website.

If a user wished to get in contact with one of the shop owners or if a shop owner wished to get in contact with a customer about an order, we provided an email service to allow this communication. As a customer, they can go to the about us page and they will find a contact form at the bottom of the page. For an admin, when viewing a customer's order, they will be able to select an option that brings them to a page with an email template and the customer details and order status autofilled. The admin can add or remove the contents of the email before it is sent.

We decided that to make site navigation easier on the customer, we added product categories which just allow a user to view specific products they wish to see rather than viewing the whole catalogue.

When creating a user, we wanted to avoid a plain text password being placed into our database, as in any real world situation this would be a serious security problem. To rectify this we used password encoding. When the password is placed into the database, it is not readable by anyone who views the database.

As a customer, we felt it was necessary to allow a user to modify their account details as people can move location, get a new phone number/email and so the website needs to accommodate these changes and allow a user to update their old details.

Finally, the final additional features we added are exclusively for admins. The specification says we must list all previous orders to the admin and we decided to simplify this by allowing the admin to instead view orders by their current status. If they wanted to only view NEW orders, then they can simply click on that button in the admin dashboard and an AJAX call is made to display those orders on the dashboard. This is a similar process if the admin wanted instead view a SHIPPED, DELIVERED or CANCELLED order. Admins can also upload new images when creating a new product to be listed on our website. This is to ensure a consistent quality across the website so that new products don't display with a no image error in contrast to the products already available which do have images.

Also our website is capable of letting multiple users interact with the website at the same time using sessions (cookies).

Req. #	Title	Comment	Status
A0	Wishlist	Integrated with HTTP Sessions	COMPLETE
A1	Product Recommendations	Displayed below each product on their viewing page	COMPLETE
A2	Contact Form/Email Service	Integrated with our own SimpleEmailService and GMail	COMPLETE
A3	Browse by Product Categories	Simple HTML pages and buttons that direct according to the category a user wants	COMPLETE
A4	Spring Security/Password Encoding	Used BCrypt Password Encoding	COMPLETE
A5	Sync Wishlists & carts to Users	Integrated with HTTP Sessions	COMPLETE
A6	Admin - Email customers with templates	Integrated with our own SimpleEmailService and GMail	COMPLETE
A7	Account details modifiable	Can be modified by going to the account page for a logged in user	COMPLETE
A8	Sort and view orders by their current state - admin	Go to the admin dashboard to sort and view orders	COMPLETE
A9	Image upload for new products and change image for old products	Integrated with our own file upload utility	COMPLETE
A10	Multiple concurrent guest users using accounts and sessions	Multiple guest users and logged in users using cookies	COMPLETE

Reflections

One of the key learnings for us from this project was that progress is not immediate - though we might have initially thought this project might be one which we could complete overnight, we quickly discovered that thorough planning and design phases would be necessary.

When planning for the project, one feature which we immediately identified as one which might be difficult to implement was the Cart and Wishlist feature. Though on the surface they seem quite simple, the objective of allowing any number of concurrent users to maintain their own carts and wishlists, where some users will be logged in and some will not, definitely presented a challenge. This represented an important learning opportunity for us, in that we gained an in-depth understanding of the operation of HTTP Sessions as well as Spring Security, which were both important components of our solution. Having completed this feature, we feel more prepared to complete similar tasks in the future.

Another feature which we identified as a potential cause for concern was the image upload feature which would need to be implemented so that administrators could add products to the site while the site was live. This represented an important learning opportunity for us in the context of file transfer over HTTP, file encoding, and also file storage.

If we were to approach this project again, one change which we would consider making is in the context of data storage. Though we used MySQL as the primary and only database for this project, we feel it might be appropriate to vary the databases used for particular pieces of data, according to the specific use cases in which that data is accessed. For example, since cart data is likely to be queried often with a minimal allowance for latency, we would consider migrating cart storage so an in-memory database such as Redis, or maybe to a managed key-value store such as Amazon Dynamo. Though database diversification can bring with it some complication, it is recommended in cases where the data access patterns are not consistent for all of the data.