Project Report

CST2120 – Coursework 2 part 2

Name: Adam Romanowski

Student Number: M00693891

Name: Rodrigo Ferreira

Student Number: M00736217

Name: Marcin Chamik

Student Number: M00732445

1. Introduction

The goal of this project is to create a fully functional e-commerce website.

2. Website Description

2.1 Front-end

**Design purpose**: Convert online sales

**Customer Area:**

Search – allow customers to input a product name and look for matches

Home – get home / landing page that aims to generate conversions

Shop - go to a page which displays all available products from the database and allows the customer to filter the results by product category, brand and price; show recommendations

About us - learn more information about the business, get about us page

Contact – display a form to be filled by the customer

MyAccount - If logged allow costumer to view page where customers can see their account details and record of orders, otherwise prompt for login/register

Login/Register - create a Customer account or login

Cart – get cart page, save a list of products, remove product, change quantity and complete purchase

**Admin/CMS Area:**

Admin login – prompt admin to authenticate

Add product - get a form for the admin to fill with new product information

Edit product - get a simple list of products available, after choosing a product get form where input fields are automatically filled by the server

View products - get a complete list of products available with all product information

Orders - get list of all orders from each customer plus their email, name and username, allow removal of orders

2.2 Back-end

**Back-end features**

Validate post requests and user input.

Use of cookies to improve customer experience for login and product recommendations by providing local authentication. Customer product views are tracked once authenticated and the information is stored in the customer data field in the database for recommendations.

AJAX for client/server communication.

HTML, CSS, Javascript for the frontend, MongoDB (noSQL) database, PHP for server side scripts running on Apache server.

Search - post request is sent to the server which returns the filtered list of matching products from the database.

**Front-end interaction**

**Costumer Area:**

Shop – Filter products by product category, brand and price

Contact - form data is posted to the server and stored in the database

MyAccount – keep track of user authentication; if authentication is passed post page with customer name, email, username and orders records

Login/Register – authenticate user, data is posted to the server and registered in the database to improve customer experience.

Cart - post request with purchase, including removal, change of quantity and complete purchase data to the server, buy button displays login/register message followed by a if successful confirmation message and stores order in the database

**Admin/CMS:**

* Admin login – post CMS login form, if authentication is approved get admin control panel to edit database
* Add product - input fields are posted to the server, new product is created and sent to the database
* Edit product – after editing post new product data to the server and store in the database
* View product – list products from database
* Orders – post edited/removed orders to the server and update the database

3. Resources

- <https://stackoverflow.com/>

- <https://github.com/>

- online lectures/labs

- Q&A sessions

- <https://www.youtube.com/>

4. Screenshots

**Home/Landing**

Graphical user interface, website

Description automatically generated

**Shop**

Graphical user interface, application, website

Description automatically generated

**About**

Text

Description automatically generated

**Contact**

Graphical user interface, application, website

Description automatically generated

**My Account**

Graphical user interface, application

Description automatically generated

**Customer login/register**

Graphical user interface, application

Description automatically generated

**Cart**

**Graphical user interface, application

Description automatically generated**

**Admin login**

Graphical user interface, text, application

Description automatically generated

**CMS**

**Graphical user interface, text, application

Description automatically generated**

5. Security, Privacy and Legal

**5.1 Security**

An e-commerce website business model would hold a large amount of customer data that may attract individuals, groups, or organisations to gain access or steal this information. We can increase security by sanitizing all user and staff input server-side (PHP and MongoDB execution) and client-side (to avoid automation, with a CAPTCHA for example) preventing SQL/noSQL injection or malicious code injection. Using a CMS we should also limit access control to the database and pages along with keeping a backup of the information to minimize damage. Cookies should remain in the same domain and have a short life span.

The use of commercial or open-source technologies allows attackers to explore the same vulnerabilities exploits that have been tested to work on the same systems and versions. We can try to prevent this by keeping our software updated and patched and avoiding backdoors.

To prevent being vulnerable to virus, malware, and other types of attacks we should backup all our data regularly and having a dedicated Intrusion Detection System and a Firewall to prevent malicious attempts. DDOS attacks can be prevented by using an IDS and web caching on a large scale.

Sensitive data like passwords and credit card information should be stored using strong encryption to discourage and try to prevent access to this data in case of a failed attack prevention

We can minimise physical attacks, social engineering, and phishing by educating the business team.

We can also prevent problems by using strict types and checking array bounds for the implementation of the project otherwise the program might be crashed while running and opening the door to another kind of vulnerabilities.

The system should be monitored by a team to increase security and response speed as well as being tested to ensure difficulty of access to bad actors.

Using a cloud service provider decreases the responsibility of ensuring security as depending on the contract part of the infrastructure, platform or application may be protected by the provider, bringing the positive factor that data is kept and transferred encrypted, but might bring other concerns related to data privacy and physical location of storage.

Most successful attacks happen because of poor configuration of the systems in use and therefore this should be taken into account.

**5.2 Privacy**

Nowadays it’s easy to track people by their online footprint. Although it's useful for governments, some people don’t agree with their privacy being ignored for safety/control reasons.

Strong encrypted services like blockchain which allow online secure payment with cryptocurrency, encrypted and safer browsers like tor, encrypted email services, and avoiding creating online accounts will increase anonymity and reduce personal footprint.

**5.3 Legal**

Some countries have started to create norms to control the movement and use of personal data online, like Europe, where you must make a legal binding contract with the user if you intend to use their personal data online and transferring data between EU and non-EU countries without legal contracts may not be recommended.

Illegal data uploaded by users must be handled as soon as it is noticed otherwise legal consequences may occur in.

In general, it’s a good idea to store the least amount of personal data needed as this will reduce the consequences of data breaches and increase user privacy. Personal data is worth money for someone somewhere so there is a constant need to keep this information secure as this can impact peoples’ lives, as long as prompt identity theft this there are legal penalties from this along with business consequences.