Reflection & Evaluation Reports

Internet Development

Submitted By: 77160000

Leeds Beckett University

BSC (Hons) Computing

Contents

[1. Guest Lecture Reflection 2](#_Toc481349425)

[2. Evaluation Report on Secure Web Development 2](#_Toc481349426)

[2.1 Web Applications and their security issues 2](#_Toc481349427)

[2.2 Evaluation Technology 2](#_Toc481349428)

[2.3 Case Study on Cross Site Scripting (XSS) 3](#_Toc481349429)

[2.5 Cross Site Scripting Prevention: 3](#_Toc481349430)

[2.6 Conclusion 4](#_Toc481349431)

[3. Evaluation Report on Web Application Optimization 4](#_Toc481349432)

[3.1 Introduction 4](#_Toc481349433)

[3.2 Overview of Web Application Optimization Techniques 4](#_Toc481349434)

[3.4 Case Study on Code Optimization 6](#_Toc481349435)

[3.3 Conclusion 6](#_Toc481349436)

[4. References 7](#_Toc481349437)

# 1. Guest Lecture Reflection

The guest lecture on digital marketing and ecommerce was led by Mr. Amit Tuladhar (muncha.com). The session begun with an outline of the E business area and the different up and coming portions in the same. And afterward went ahead to address Digital Marketing-its need and significance in the present promoting situation. Illuminated the part of multi-divert E-Commerce in Digital advertising. It clarified how a web search tool functions and what the sorts of ventures included are. Additionally highlighted the significance and importance of "push" and draw" advertising in advanced promoting. How online networking functions and is adequacy in computerized showcasing was talked about. Significance and ideal models of Web Analytics were additionally touched upon towards the conclusion. The session got intuitive towards the end and you could see the enthusiasm of the understudies as they could identify with the substance of the visitor address as the case from the day today life helped them relate with the point.

# 2. Evaluation Report on Secure Web Development

## **2.1 Web Applications and their security issues**

Web applications might be characterized as PC projects that are gotten to over a network, for example, the Internet or an intranet. Web applications (normally called websites) are the human interfaces to on-line exercises (i.e. on-line retail deals, webmail, social communities) gotten to through web browsers, for example, Internet Explorer, Firefox, Chrome and so on (Wlasuk, 2012).

Web application security issues emerge when the web application responds inadequately to unexpected data demands or uniquely made information from a client with a malicious aim. A confounded web application may turn into a security hazard when that perplexity permits unintended control of the web application usefulness itself (Wlasuk, 2012).

For instance, consider an on-line retail deals web application where a client, via browser interactions, trades information with a web application that is facilitated inside the company's data center. The association between the client and the data center facilitated web application has, by need, skirted all physical security. To compound potential security issues, the web application overhauling the client will frequently have guide access to the database containing retail exchange data for all clients of that on-line deals application. The main protect between the client and the move database is the web application (Wlasuk, 2012).

## **2.2 Evaluation Technology**

**2.2.1 Logging and Error Handling:**

This segment manages outlining elegantly composed applications that have double reason logs and action follows for review and checking. This makes it simple to track an exchange without over the top exertion or access to the framework. They ought to have the capacity to effectively track or recognize potential misrepresentation or inconsistencies end-to-end (it.northwestern.edu, 2016).

**2.2.2 Data Validation:**

This segment manages applications being powerful against all types of info information, regardless of whether gotten from the client, framework, outside elements or databases.

**2.2.3 Authorization:**

This area addresses validation issues, guaranteeing a client has the suitable benefits to see an asset. Themes, for example, rule of minimum benefit, customer side approval tokens, and so forth are tended to here.

**2.2.4 Authentication:**

This area manages verification issues related with secure web applications, for example, fundamental/process validation, frame based confirmation, incorporated (SSO) verification, and so forth.

**2.2.5 Session Management:**

This segment addresses points, for example, confirmed clients having a strong and cryptographically secure relationship with their session, applications upholding approval checks and applications keeping away from or anticipating normal web assaults, for example, replay, ask for manufacturing and man-in-the-center (it.northwestern.edu, 2016).

**2.2.6 Cryptography:**

This segment serves to guarantees that cryptography is securely used to ensure the privacy and uprightness of delicate client information

## **2.3 Case Study on Cross Site Scripting (XSS)**

Cross Site Scripting is a scope of utilization level assaults. It is brought about by the disappointment of a sufficient approval before returning site pages to the customer's program. Clients submit malevolent HTML to dynamic web applications. At the point when different clients see the pernicious substance it seems to originate from a confided in source. The malevolent HTML might be implanted inside URL parameters, shapes fields and treats. Clients input that incorporates pernicious code will be executed by the customer's program unbeknownst to the client, creating a security assault. In a web application sign on two confirmation tokens, for example, a username and secret key is traded for a treat. Since those qualities are put away in a treat that a client session is defenseless if caught by the cross site scripting assault. With this treat the aggressor can stack it, indicate the program the relating site, and get to the casualties account. The degree of this effect relies on upon the security components of the web application. For instance if there is stored or spared Visa data this programmer can purchase things and may even get to bank records.

This is a genuinely across the board input sanitization failure like SQL injection. An attackers send JavaScript tags on input to your web application. At the point when this info is come back to the client unsanitized, the client's program will execute it. It can be as straightforward as making a connection and inducing a client to snap it, or it can be something substantially more evil. On page stack the script runs and, for instance, can be utilized to present your treats on the assailant (KALMAN, 2017).

## **2.5 Cross Site Scripting Prevention:**

There's a straightforward web security arrangement: don't return HTML labels to the customer. This has the additional advantage of safeguarding against HTML infusion, a comparable assault whereby the aggressor infuses plain HTML substance like pictures or noisy imperceptible glimmer players not high-affect but rather most likely irritating ("please make it stop!"). Ordinarily, the workaround is just changing over all HTML substances, so that <script> is returned as &lt;script&gt;. The other frequently utilized technique for disinfection is utilizing consistent expressions to strip away HTML labels utilizing standard expressions on < and >, yet this is hazardous as a great deal of programs will decipher extremely broken HTML fine and dandy. Better to change over all characters to their got away partners (KALMAN, 2017).

## **2.6 Conclusion**

I trust that I have figured out how to stimulate your mind a tiny bit with this post and to present a solid measurement of suspicion and web security powerlessness mindfulness. The center takeaway here is that deep rooted programming hones exist for a reason and what connected some time ago for cushion floods, still apply for salted strings in Python today. Security helps you compose correct programs, which all software engineers ought to hope for.

# 3. Evaluation Report on Web Application Optimization

## **3.1 Introduction**

Now a days, Web applications are the core of each and every business’s corporations all over the world. Web optimization is the serious portion of web application development and maintenance. It is the reason behind driving revenue and losing significant profit. Web application means a software which is design and developed, supported by any browser that any user can suffers in their wished web browser. A web application contains different interconnected modules and components which is always work together for processing the data and finally execute outputs. Optimization helps to improve weak performance when you find while using web application. Optimization also help to improve loading times for website pages. With the help of optimization you can handle the data transfer rate according to user desires. I also help to reduce needless processing time. Optimization takes into account the calibrating of all these web application segments with a specific end goal to help make it run quicker, smoother, and simpler. This, thusly, upgrades client experience and gives clients motivation to return to the web application and in addition impart it to their contacts, boosting brand perceive ability and navigate rates and that by and large means enhanced business execution and higher benefits.

## **3.2** **Overview of Web Application Optimization Techniques**

There are many variety of strategies for optimization in a web application. And they all are covenant with two main key layers which are the application layer and the presentation layer. You can find in details about these two main layers of optimization below.

**3.2.1 Application Layer**

The application layer handles the UI of the web application, guaranteeing that the vital assets exist for effective communication with other application programs over a system. This likewise incorporates guaranteeing the recognizable proof of the goal have, verifying either or both the message sender and beneficiary, deciding convention manages, and guaranteeing understanding at both closures about mistake information uprightness, recuperation methodology, and security. Dealing with these segments can help your application stack quicker and perform better from a convenience point of view (ServerSide, 2015).

**3.2.2 Database Optimization**

Inside the application layer, guarantee that all database-related components work well. Any shortcomings at this level may affect client experience and Google page rank, and additionally back off quick running inquiries in different parts of the server. Three strategies you can use to lift database improvement include:

* **Indexing**: A key strategy to build question execution is the formation of productive records, which diminishes the measure of information that an inquiry needs to handle by enhancing the effectiveness of read operations. There are two sorts of records, bunched and non-grouped, each of which has its own preferences relying upon the informational collection. A grouped file is more proficient on sections of information while a non-bunched record is more qualified for inquiries bringing about correct matches (ServerSide, 2015).
* **Table Partitioning**: parts one extensive table into littler ones, enhancing inquiry execution, lessening list size, and rendering it more probable that the intensely utilized parts of the records fit in memory (ServerSide, 2015).
* **Enabling Query Caching**; or the storing of usually utilized SQL inquiries in memory, is an awesome approach to accelerate the running of your web application. Inquiry storing empowers the snappy return of results for frequently utilized inquiries, which is particularly valuable for generally sites (ServerSide, 2015).

**3.2.3 Application Server Optimization**

Web applications are facilitated and made accessible to clients by means of the application server. Advancing the application server guarantees that the application may work speedier. Two procedures for streamlining the execution of the application server include:

* **Code Caching**: putting away a few or the majority of the data your code produces in a record and returning it when required. This strategy guarantees that a quick show of substance, particularly in web situations where a similar substance is shown again and again to guests (ServerSide, 2015).
* **Code Refactoring**: elucidating and improving the outline of existing code by exchanging inside code and not its outer usefulness. This method makes it less demanding to keep up existing code while growing new functionalities meaning more productive executions, simpler updates, and speedier download speeds (ServerSide, 2015).

**3.2.4 Presentation Layer**

The introduction layer handles the data that is sent to the client, guaranteeing that every one of the information is in the right arrangement and is without errors. Two valuable procedures for improving the introduction layer of web applications are reserve controlling and resource enhancement (PRICE, 2012).

**3.2.5 Cache Controlling**

Controlling headers on substance is basic to the execution of a web application, boosting application responsiveness while decreasing system movement and superfluous transfer speed utilization. There are three sorts of store to remember: program reserves, intermediary reserves, and passage reserves.

**3.2.6 Asset Optimization**

Improving the execution of web segments, including pictures and static scripts, diminishes asset demands and the apparent load time for the client. Take in more about resource improvement here and here (ServerSide, 2015).

## **3.4 Case Study on Code Optimization**

**3.4.1 Data Compression**

This section contains encoding data using fewer bits, eliminating unnecessary data always yields the best results. For best data compression you have to achieve different types of compression techniques.

**3.4.2 Minification: preprocessing and context specific optimizations**

This section contains reduction of the size of delivery resources like reduces network payload. It also help to remove unnecessary data of html, JS and CSS like whitespace, comments. There are lots of optimizing tool like for html and CSS compressor there is an online tool which URL is <https://htmlcompressor.com/compressor/> . You can compressor code with this tool online.

**3.4.3 Text Compression with GZIP**

Gzip is a strategy for packing documents (making them littler) for quicker system exchanges. It is additionally a record design. Pressure enables your web server to give littler document sizes which stack speedier for your site clients. Empowering gzip pressure is a standard practice

## **3.3 Conclusion**

Web application optimization is critical, and today's Web application advancement group can help webpage execution and enhance its website's heap trying scores by choosing from an assortment of customer side reserving procedures. A viable customer side storing methodology can decrease stack times by a few elements. The latest developments in customer side reserving, for example, the HTML5 Application Cache, empower an application to run (however maybe in a more constrained frame) even without a system association introduce (PRICE, 2012).

# 4. References

it.northwestern.edu, 2016. *Guide to Securing Web Applications.* [Online]   
Available at: http://www.it.northwestern.edu/policies/webapps.html  
[Accessed 30 4 2017].

KALMAN, G., 2017. *10 Most Common Web Security Vulnerabilities.* [Online]   
Available at: https://www.toptal.com/security/10-most-common-web-security-vulnerabilities  
[Accessed 27 4 2017].

PRICE, S., 2012. *Web Performance Optimization, Part 10: Client Side Caching.* [Online]   
Available at: https://loadstorm.com/2012/02/web-performance-optimization-evolution-client-side-caching/  
[Accessed 28 4 2017].

ServerSide, 2015. *Optimizing the Performance of Web Applications.* [Online]   
Available at: https://www.serverside.net/blog/optimizing-the-performance-of-web-applications/  
[Accessed 27 4 2017].

Wlasuk, A., 2012. *Case Study: Securing Web Applications.* [Online]   
Available at: http://www.securityweek.com/case-study-securing-web-applications  
[Accessed 27 4 2017].