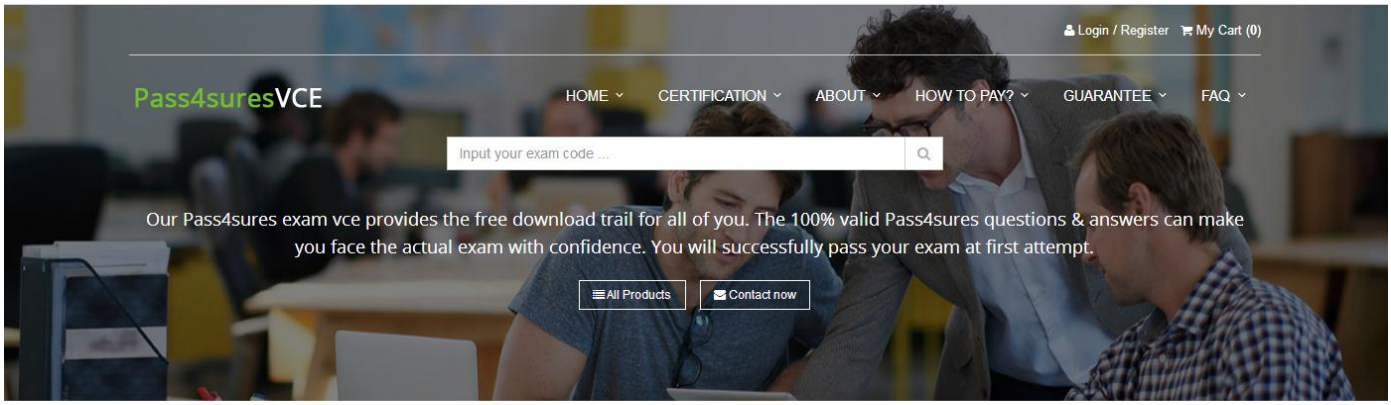
Pass4suresVCE



QUALITY AND VALUE

Pass4suresVCE Practice Exams are written to the highest standards of technical accuracy, using only certified subject matter experts and published authors for development - no all dumps.

TESTED AND APPROVED

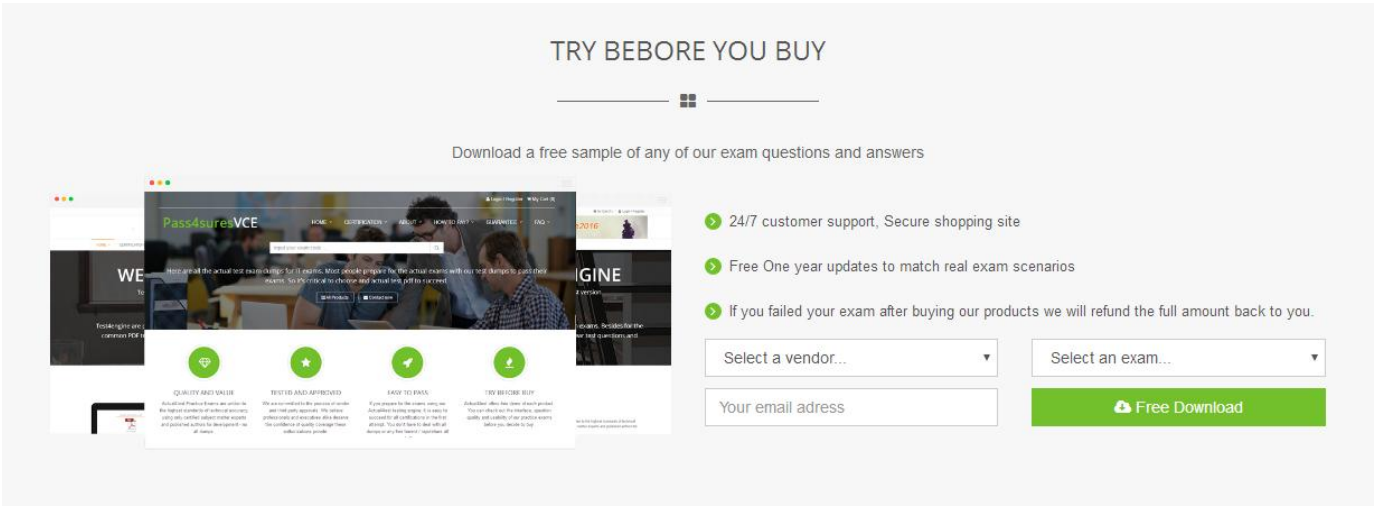
We are committed to the process of vendor and third party approvals. We believe professionals and executives alike deserve the confidence of quality coverage these authorizations provide.

EASY TO PASS

If you prepare for the exams using our Pass4suresVCE testing engine, It is easy to succeed for all certifications in the first attempt. You don't have to deal with all dumps or any free torrent / rapidshare all stuff.

TRY BEFORE BUY

Pass4suresVCE offers free demo of each product. You can check out the interface, question quality and usability of our practice exams before you decide to buy.



http://www.pass4suresvce.com

Pass4sures exam vce dumps for guaranteed success with high scores

Exam : **312-96**

Title : Certified Application Security Engineer (CASE) JAVA

Vendor : ECCouncil

Version : DEMO

NO.1 Suppose there is a productList.jsp page, which displays the list of products from the database for the requested product category. The product category comes as a request parameter value. Which of the following line of code will you use to strictly validate request parameter value before processing it for execution?

A. public boolean validateUserName() {String CategoryId= request.getParameter("CatId");}

B. public boolean validateUserName() { Pattern p = Pattern.compile("[a-zA-Z0-9]*$"); Matcher m = p.matcher(request.getParameter(CatId")); boolean result = m.matches(); return result;}

C. public boolean validateUserName() { if(request.getParameter("CatId")!=null) String CategoryId=request.getParameter("CatId");}

D. public.boolean validateUserName() { if(!request.getParamcter("CatId").equals("null"))}

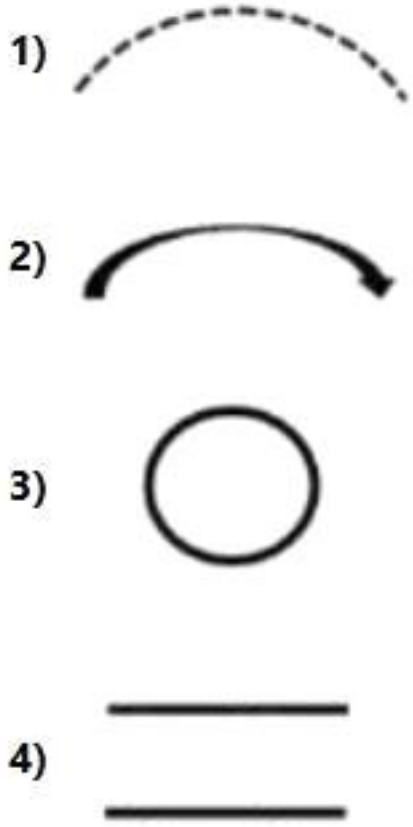
Answer: B

Explanation:

The correct line of code for strictly validating the request parameter value before processing it for execution is option B. This code snippet uses a regular expression to ensure that the CatId parameter only contains alphanumeric characters, which is a common validation technique to prevent SQL injection and other forms of attacks. The Pattern.compile("[a-zA-Z0-9]*$") creates a pattern that matches a string of zero or more alphanumeric characters. The matcher method is then used to match the pattern against the CatId parameter obtained from the request. If the parameter matches the pattern, m.matches() returns true, indicating that the parameter is valid.

References: The answer provided is in accordance with the best practices for input validation as outlined in the EC-Council's Certified Application Security Engineer (CASE) JAVA training and certification program. The program emphasizes secure coding practices, including input validation to protect against common security threats such as SQL injection. For further details, please refer to the official EC-Council CASE JAVA course materials and study guides12.

NO.2 Which of the following DFD component is used to represent the change in privilege levels?



A. 3
B. 4
C. 1
D. 2
Answer: D
Explanation:
In a DFD, different components represent different aspects of the system:
* Circles or ovals usually represent processes or functions where data is processed or transformed.
* Arrows represent data flows moving from one part of the system to another.
* Open rectangles represent external entities or actors that interact with the system.
* Parallel lines represent data stores or repositories where data is held.
Given these conventions, a change in privilege levels would most likely be associated with a process, as it involves a transformation or decision within the system. Therefore, the component that represents a process (typically a circle or oval) would be used to depict a change in privilege levels.
References: For accurate and verified answers, please refer to the official EC-Council Application Security Engineer (CASE) JAVA study guides and course materials12. These resources will provide the most reliable information regarding the specifics of DFD components as they pertain to the CASE JAVA certification.

NO.3 Which of the following Spring Security Framework configuration setting will ensure the protection from session fixation attacks by not allowing authenticated user to login again?
A. session-fixation-protection ="newSessionID"
B. session-fixation-protection ="".

C. session-fixation-protection ="enabled"

D. session-fixation-protection =".

Answer: B

Explanation:

Spring Security provides built-in protection against session fixation attacks. It does this by invalidating the existing session and creating a new one when a user authenticates. This behavior can be configured using the sessionManagement() method in the Java configuration. The newSession strategy, which is the default, changes the session ID upon authentication to protect against session fixation.

Here's an example of how it can be configured:

Java

http.sessionManagement()

sessionFixation().migrateSession();

AI-generated code. Review and use carefully. More info on FAQ.

This configuration ensures that a new session is created, and the old one is invalidated when the user logs in, thus providing protection against session fixation attacks.

References:The information provided is based on the standard configuration practices for Spring Security to protect against session fixation attacks. For more detailed information, you can refer to the official Spring Security documentation123 and other authoritative resources on Spring Security session management.

NO.4 Which of the risk assessment model is used to rate the threats-based risk to the application during threat modeling process?

A. DREAD

B. SMART

C. STRIDE

D. RED

Answer: C

Explanation:

STRIDE is a risk assessment model used to identify and rate threats during the threat modeling process. It stands for Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. Each element of STRIDE represents a specific type of threat that could potentially affect an application, and it is used to systematically assess the security of an application by identifying possible vulnerabilities that could be exploited by these threats.

References: The EC-Council's Certified Application Security Engineer (CASE) JAVA course emphasizes the importance of threat modeling in the software development lifecycle and specifically mentions the use of models like STRIDE to assess and mitigate risks12.

NO.5 Alice, a security engineer, was performing security testing on the application. He found that users can view the website structure and file names. As per the standard security practices, this can pose a serious security risk as attackers can access hidden script files in your directory. Which of the following will mitigate the above security risk?

A. < int-param > < param-name>directory-listinqs < param-value>true < /init-param >

B. < int param > < param-name>directorv-listinqs < param-value>false < /init-param >

C. < int-param > < param-name>listinqs < param-value>true < /init-param

D. < int-param > < param-name>listinqs < param-value>false < /init-param >

Answer: D

Explanation:

To mitigate the security risk of users being able to view the website structure and file names, the correct action would be to disable directory listings. This is often accomplished through configuration settings in web server software, where you can specify whether to allow or deny the listing of directory contents. The option < int-param> <param-name>listings <param-value>false</int-param> effectively disables directory listings, preventing users and potential attackers from viewing the website's file and directory structure, thus enhancing security. Ensuring that directory listings are disabled is a common security practice to avoid revealing sensitive information about the web application's structure.References:

* Web Server Security Best Practices documentation
* OWASP (Open Web Application Security Project) guidelines on securing web server configurations