High Hit Rate Valid CRT-450 Cram Materials–Find Shortcut to Pass CRT-450 Exam

DOWNLOAD the newest ActualVCE CRT-450 PDF dumps from Cloud Storage for free: https://drive.google.com/open?id=1h1T_7uQGVRrXmcdEJYPkzF1qZD1jFC16

It is well known, to get the general respect of the community needs to be achieved by acquiring knowledge, and a harvest. Society will never welcome lazy people, and luck will never come to those who do not. We must continue to pursue own life value, such as get the test CRT-450 Certification, not only to meet what we have now, but also to constantly challenge and try something new and meaningful.

Salesforce CRT-450 Exam is a challenging but rewarding certification that is highly valued in the Salesforce community. It is an excellent way for individuals to demonstrate their expertise in Salesforce development and stand out from their peers in the industry.

>> Valid CRT-450 Cram Materials <<

CRT-450 Reliable Exam Question & CRT-450 Latest Test Prep

The high quality of our CRT-450 preparation materials is mainly reflected in the high pass rate, because we deeply know that the pass rate is the most important. As is well known to us, our passing rate has been high; 99% of people who used our CRT-450 real test has passed their tests and get the certificates. I dare to make a bet that you will not be exceptional. Your test pass rate is going to reach more than 99% if you are willing to use our CRT-450 Study Materials with a high quality. So it is necessary for you to

know well about our CRT-450 test prep.

Salesforce Certified Platform Developer I Sample Questions (Q120-Q125):

NEW QUESTION # 120
A developer completed modifications feature that is comprised of two elements:
* Apex trigger
* Trigger handler Apex class
What are two factors that the developer must take into account to properly deploy them to the production environment?
Choose 2 answers

- A. All methods in the test classes must use @istest.
- B. Test methods must be declared with the testMethod keyword.
- C. Apex classes must have at least 75% code coverage org-wide.
- D. At least one line of code must be executed for the Apex trigger.

Answer: C,D

Explanation:
75% Code Coverage (A):To deploy Apex code to production, Salesforce requires at least 75% of the code to be covered by tests across the organization.
Reference:Apex Testing Framework
Trigger Coverage (C):Apex triggers must have at least one line of code covered by test execution to deploy.
Reference:Testing Apex Triggers
Incorrect Options:
B:Although@isTestis recommended, it is not mandatory for all methods.
D:ThetestMethodkeyword is outdated and replaced by@isTest.

NEW QUESTION # 121
Assuming that name is a String obtained by a Visualforce page, which two SOQL queries performed are safe from SOQL injection?
(Choose two.)

- A. apex
  Copy
  String query = '%' + name + '%';
  List<Account> results = [SELECT Id FROM Account WHERE Name LIKE :query];
- B. apex
  Copy
  String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name.noQuotes() + '%\''; List<Account> results = Database.query(query);
- C. apex
  Copy
  String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name + '%\''; List<Account> results = Database.query(query);
- D. apex
  Copy
  String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + String.escapeSingleQuotes (name) + '%\'';
  List<Account> results = Database.query(query);

Answer: A,D

Explanation:
Comprehensive and Detailed Explanation From Exact Extract:
To determine which SOQL queries are safe from SOQL injection, we need to evaluate each option for how it handles the name parameter (user input from a Visualforce page) and whether it properly mitigates the risk of SOQL injection. SOQL injection occurs when untrusted user input is directly embedded into a query string, allowing malicious users to manipulate the query's logic. Let's analyze each option systematically, referencing Salesforce's official documentation, particularly the Secure Coding Guidelines and Apex Developer Guide.
Understanding SOQL Injection:
* SOQL Injection: This vulnerability arises when user input is dynamically concatenated into a SOQL query string without proper

sanitization, allowing an attacker to alter the query's behavior. For example, if name is ' OR '1'='1, an unsafe query might return all records instead of the intended subset.
The Salesforce Secure Coding Guidelines state: "SOQL injection occurs when untrusted input is concatenated into a query string, potentially allowing an attacker to bypass intended logic" (Salesforce Secure Coding Guidelines, SOQL Injection).
* Prevention Techniques:
* Bind Variables (:variable): Using bind variables in SOQL queries ensures user input is treated as a value, not part of the query logic, preventing injection. The Apex Developer Guide notes:
"Using bind variables (:variable) in SOQL queries is the safest way to prevent SOQL injection" (Salesforce Apex Developer Guide, Secure Coding for SOQL).
* Sanitization: If dynamic SOQL (e.g., Database.query()) is used, user input must be sanitized using methods like String.escapeSingleQuotes() to escape special characters (e.g., single quotes) that could alter the query.
* Context: The name variable comes from a Visualforce page, meaning it's untrusted user input and must be handled carefully to prevent injection.
Evaluating the Options:
* A.
apex
Copy
String query = '%' + name + '%';
List<Account> results = [SELECT Id FROM Account WHERE Name LIKE :query];
* Approach: Constructs a query string by concatenating wildcards (%) with the name variable, then uses a bind variable (:query) in the SOQL query.
* Security: Using a bind variable (:query) ensures that the value of query (which includes name) is treated as a literal value in the LIKE clause, not as part of the query's syntax. Even if name contains malicious input (e.g., ' OR '1'='1), the SOQL engine treats it as a single string to match against Name, preventing injection. The Apex Developer Guide confirms: "Bind variables prevent SOQL injection by treating user input as data, not executable code" (Salesforce Apex Developer Guide, Secure Coding for SOQL).
* Example: If name = 'Test' OR '1'='1, then query = '%Test' OR '1'='1%'. The SOQL query becomes:
apex
Copy
[SELECT Id FROM Account WHERE Name LIKE '%Test\' OR \'1\'=\'1%']
This searches for records where Name matches the literal string %Test' OR '1'='1%, which is safe and does not alter the query logic.
* Conclusion: Safe from SOQL injection due to the use of a bind variable.
* B.
apex
Copy
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name.noQuotes() + '%\''; List<Account> results = Database.query(query);
* Approach: Dynamically constructs a SOQL query string by concatenating name.noQuotes() into the query, then executes it using Database.query().
* Security: The noQuotes() method is not a standard Apex method on the String class. The Apex Developer Guide does not define noQuotes() as a built-in method (Salesforce Apex Developer Guide, String Class). Assuming it's a custom method, it likely attempts to remove quotes from the string, but this does not prevent SOQL injection. Concatenating user input directly into the query string is inherently unsafe unless properly sanitized.
* Example: If name = 'Test' OR '1'='1, and assuming noQuotes() does nothing (or removes quotes, which doesn't help), the query becomes:
apex
Copy
SELECT Id FROM Account WHERE Name LIKE '%Test' OR '1'='1%'
The OR '1'='1' condition evaluates to true for all records, returning all Accounts, which is a successful SOQL injection attack.
* Conclusion: Not safe, as it directly concatenates user input without proper sanitization or bind variables, and noQuotes() is not a reliable or standard method for preventing injection.
* C.
apex
Copy
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + String.escapeSingleQuotes(name) +
'%\'';
List<Account> results = Database.query(query);
* Approach: Dynamically constructs a SOQL query string by concatenating name into the query after applying String.escapeSingleQuotes(), then executes it using Database.query().
* Security: The String.escapeSingleQuotes() method escapes single quotes in the input by adding a backslash (\), preventing the input from breaking out of the string literal in the SOQL query. The Apex Developer Guide states: "String.escapeSingleQuotes()

prevents SOQL injection in dynamic queries by escaping single quotes, ensuring the input cannot alter the query structure" (Salesforce Apex Developer Guide, String Class). This ensures that even malicious input cannot manipulate the query logic.
* Example: If name = 'Test' OR '1'='1, then String.escapeSingleQuotes(name) returns Test\' OR \'1\'=\'1. The query becomes:
apex
Copy
SELECT Id FROM Account WHERE Name LIKE '%Test\' OR \'1\'=\'1%'
This searches for records where Name matches the literal string %Test' OR '1'='1%, which is safe and does not allow the OR condition to execute as logic.
* Conclusion: Safe from SOQL injection due to the use of String.escapeSingleQuotes() to sanitize the user input in a dynamic query.
* D.
apex
Copy
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name + '%\''; List<Account> results = Database.query(query);
* Approach: Dynamically constructs a SOQL query string by directly concatenating name into the query, then executes it using Database.query().
* Security: This approach is vulnerable to SOQL injection because name is directly embedded into the query string without sanitization. The Salesforce Secure Coding Guidelines warn: "Directly concatenating user input into a dynamic SOQL query without escaping can lead to SOQL injection" (Salesforce Secure Coding Guidelines, SOQL Injection).
* Example: If name = 'Test' OR '1'='1, the query becomes:
apex
Copy
SELECT Id FROM Account WHERE Name LIKE '%Test' OR '1'='1%'
The OR '1'='1' condition evaluates to true for all records, returning all Accounts, demonstrating a successful SOQL injection attack.
* Conclusion: Not safe, as it directly concatenates user input without sanitization or bind variables.
Why Options A and C are Correct:
* Option A: Uses a bind variable (:query) in the SOQL query, ensuring that the user input (name) is treated as a literal value, not executable code. This is the most secure way to prevent SOQL injection, as recommended by Salesforce.
* Option C: Uses String.escapeSingleQuotes() to sanitize the user input before embedding it into a dynamic SOQL query, preventing the input from altering the query's logic. This is a safe approach for dynamic queries when bind variables cannot be used directly.
* Options B and D: Both concatenate user input directly into the query string without adequate sanitization (noQuotes() is not a standard or reliable method in B, and D has no sanitization), making them vulnerable to SOQL injection.
Example of Vulnerability (Option D):
Consider a Visualforce page where name is set via a user input field:
apex
Copy
String name = ApexPages.currentPage().getParameters().get('name');
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name + '%\''; List<Account> results = Database.query(query); If a malicious user submits name = ' OR '1'='1, the query becomes:
apex
Copy
SELECT Id FROM Account WHERE Name LIKE '%' OR '1'='1%'
This returns all Accounts, bypassing the intended filtering, which is a successful SOQL injection attack.
Mitigating SOQL Injection:
* Preferred (Option A): Use bind variables whenever possible:
apex
Copy
String query = '%' + name + '%';
List<Account> results = [SELECT Id FROM Account WHERE Name LIKE :query];
* Alternative (Option C): If dynamic SOQL is required, sanitize input with String.escapeSingleQuotes():
apex
Copy
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + String.escapeSingleQuotes(name) + '%\'';
List<Account> results = Database.query(query);
Handling Typos:
* The options are syntactically correct in the provided image, with no typos to address.
* Option B's name.noQuotes() is not a standard Apex method, but the analysis assumes it's a custom method that fails to prevent injection, as it's not a recognized sanitization technique.

References:
Salesforce Apex Developer Guide:
"Secure Coding for SOQL" section: Recommends using bind variables to prevent SOQL injection.
"String Class" section: Details String.escapeSingleQuotes() for sanitizing dynamic queries.
"Database Class" section: Describes Database.query() for dynamic SOQL execution.(Available at:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/) Salesforce Secure Coding Guidelines:
"SOQL Injection" section: Warns against concatenating untrusted input and recommends bind variables or sanitization.(Available at:
https://developer.salesforce.com/docs/atlas.en-us.secure_coding_guide.meta
/secure_coding_guide/)
Platform Developer I Study Guide:
Section on "Salesforce Platform and Declarative Features": Covers secure coding practices, including preventing SOQL injection in Visualforce contexts.(Available at: https://trailhead.salesforce.com/en/content
/learn/modules/platform-developer-i-certification-study-guide)

NEW QUESTION # 122
What is the result of the following code snippet?
public word doWork(Account acct) {
for (Integer i = 0; i <= 2007 i++) {
insert acct;
}

- A. 201 Accounts are inserted.
- B. Account is inserted.
- C. Accounts are inserted.
- D. 200 Accounts are inserted.

Answer: A

Explanation:
ThedoWorkmethod inserts an account for every iteration of theforloop, which runs fromi = 0toi <= 200, inclusive.
This results in 201 iterations, and hence 201 accounts are inserted into the org.
Explanation of Code Behavior:
The loop conditioni <= 200runs 201 times (0 to 200).
Theinsert acct;statement is executed 201 times.
Apex Developer Guide - Loop Constructs

NEW QUESTION # 123
What is a capability of a StandardSetController?Choose 2 answers

- A. It allows pages to perform pagination with large record sets
- B. It enforces field-level security when reading large record sets
- C. It allows pages to performmass updates of records
- D. It extends the functionality of a standard or custom controller

Answer: A,C

NEW QUESTION # 124
As part of new feature development, a developer is asked to build a responsive application capable of responding to touch events, that will be executed on stateful clients.
Which two technologies are built on a framework that fully supports the business requirement? Choose 2 answers

- A. Aura Components
- B. Vlsualforce Components
- C. Lightning Web Components
- D. Visualforce Pages

Answer: A,C

Explanation:
Aura Components and Lightning Web Components are two technologies that are built on a framework that fully supports the business requirement of building a responsive application capable of responding to touch events, that will be executed on stateful clients. Both technologies are part of the Lightning Component Framework, which is a modern UI framework for developing dynamic web apps for mobile and desktop devices. The Lightning Component Framework uses standard web technologies, such as HTML, CSS, JavaScript, and Web Components, to create reusable and interoperable components that can adapt to different screen sizes, devices, and orientations. The framework also provides features such as data binding, event handling, state management, and server-side integration, to enable developers to create rich and interactive user interfaces.
Aura Components are the original technology for creating Lightning components. They use a custom XML-based markup language, Aura, to define the component structure and behavior. Aura Components can respond to touch events using the ui:input component, which provides a generic input element that can handle different input types, such as text, number, date, checkbox, radio, toggle, and email. The ui:input component also supports touch gestures, such as swipe, tap, and pinch, by using the ontouchstart, ontouchend, ontouchmove, and ontouchcancel attributes. Aura Components are stateful, meaning that they maintain their state on the client-side and communicate with the server only when necessary. This reduces the network traffic and improves the performance and user experience.
Lightning Web Components are the newer technology for creating Lightning components. They use standard HTML, CSS, and JavaScript to define the component structure and behavior. Lightning Web Components can respond to touch events using the standard HTML input element, which provides a native input element that can handle different input types, such as text, number, date, checkbox, radio, toggle, and email. The input element also supports touch gestures, such as swipe, tap, and pinch, by using the standard touch event listeners, such as touchstart, touchend, touchmove, and touchcancel. Lightning Web Components are also stateful, meaning that they maintain their state on the client-side and communicate with the server only when necessary. This reduces the network traffic and improves the performance and user experience.
Visualforce Components and Visualforce Pages are two technologies that are not built on a framework that fully supports the business requirement of building a responsive application capable of responding to touch events, that will be executed on stateful clients. Visualforce Components are reusable UI elements that can be used in Visualforce Pages. Visualforce Pages are web pages that can display and interact with Salesforce data and logic. Visualforce Components and Pages use a custom XML-based markup language, Visualforce, to define the UI elements and behavior. Visualforce Components and Pages can be made responsive by using the Salesforce Lightning Design System (SLDS), which is a collection of design guidelines, components, and resources that enable developers to create consistent and beautiful user interfaces across devices. However, Visualforce Components and Pages do not have native support for touch events, and require custom JavaScript code to handle them. Visualforce Components and Pages are also stateless, meaning that they do not maintain their state on the client-side and communicate with the server on every request. This increases the network traffic and affects the performance and user experience.
References:
* Lightning Component Framework
* Aura Components Developer Guide
* Lightning Web Components Developer Guide
* Visualforce Developer Guide
* Salesforce Lightning Design System

NEW QUESTION # 125
.....

When you have a lot of eletronic devices, you definitly will figure out the way to study and prepare your CRT-450 exam with them. It is so cool even to think about it. As we all know that the electronic equipment provides the convenience out of your imagination.With our APP online version of our CRT-450practice materials, your attempt will come true. Our CRT-450 exam dumps can be quickly downloaded to the eletronic devices.

Registration

- Real and Error Free Pdfvce Salesforce CRT-450 Exam Practice Test □ Go to website □ www.pdfvce.com □ open and search for [CRT-450] to download for free □CRT-450 Actual Questions
- CRT-450 Latest Learning Materials □ CRT-450 New Braindumps Pdf □ New CRT-450 Test Experience □ Search for { CRT-450 } and easily obtain a free download on ☀ www.exam4labs.com □☀□ □Reliable CRT-450 Dumps Ebook
- CRT-450 New Braindumps Pdf □ CRT-450 New Braindumps Pdf □ CRT-450 Valid Exam Forum □ Search for ☀ CRT-450 □☀□ and download it for free immediately on ➤ www.pdfvce.com □ □Reliable CRT-450 Dumps Ebook
- Free PDF Quiz CRT-450 - Salesforce Certified Platform Developer I Accurate Valid Cram Materials ✉ Search for ✔ CRT-450 □✔□ and download it for free immediately on [www.prep4sures.top] □Test CRT-450 Collection
- CRT-450 real exam questions, CRT-450 test dumps vce pdf □ Simply search for 《 CRT-450 》 for free download on " www.pdfvce.com " □New CRT-450 Test Experience
- Free Salesforce Certified Platform Developer I vce dumps - latest CRT-450 examcollection dumps □ Search for ➽ CRT-450 □ and download it for free immediately on □ www.easy4engine.com □ □CRT-450 New Braindumps Pdf
- www.stes.tyc.edu.tw, rhinotech.cc:88, www.stes.tyc.edu.tw, www.posteezy.com, hi-wot.com, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, wanderlog.com, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, pct.edu.pk, Disposable vapes

BONUS!!! Download part of ActualVCE CRT-450 dumps for free: https://drive.google.com/open?id=1h1T_7uQGVRrXmcdEJYPkzF1qZD1jFC16