ISQI CTAL-TAE Exam | Reliable CTAL-TAE Exam Blueprint - Sample Download Free of New CTAL-TAE Practice Materials

BONUS!!! Download part of PassLeader CTAL-TAE dumps for free: https://drive.google.com/open?id=1eqPhy4Qy5KfXP-_2QJBg8cDjr4aKdbSi

We want to finish long term objectives through customer satisfaction and we have achieved it already by our excellent CTAL-TAE exam questions. In this era of cut throat competition, we are successful than other competitors. What is more, we offer customer services 24/7. Even if you fail the exams, the customer will be reimbursed for any loss or damage after buying our CTAL-TAE Guide dump. One decision will automatically lead to another decision, we believe our CTAL-TAE guide dump will make you fall in love with our products and become regular buyers.

ISQI CTAL-TAE Exam Syllabus Topics:

Topic	Details
Topic 1	• Ability to trace the generated tests back to the model • Explaintherole that layers play within a TAA
Topic 2	• Relate test cases to test objectives or SUT requirements • Configure and parameterize the test setup
Topic 3	• Set up and tear down the SUT for test execution • Design the appropriateTAAfor a given project
Topic 4	• Specify test sequences or fully-fledged test behaviors • Analyze factors of implementation, use, and maintenancerequirementsfor a given TAS
Topic 5	• Defining test scripts for the execution of the test case • Set up and tear down test suites
Topic 6	• Developing, capturing, or deriving test data • Providing access to test libraries as needed

>> Reliable CTAL-TAE Exam Blueprint <<

Quiz High Pass-Rate ISQI - Reliable CTAL-TAE Exam Blueprint

PassLeader also offers up to 1 year of free updates. It means if you download our actual CTAL-TAE exam questions today, you

can get instant and free updates of these CTAL-TAE questions. With this amazing offer, you don't have to worry about updates in the ISTQB Certified Tester Advanced Level, Test Automation Engineering (CTAL-TAE) examination content for up to 1 year. In case of any update within three months, you can get free CTAL-TAE exam questions updates from PassLeader.

The ISTQB Certified Tester Advanced Level, Test Automation Engineering (CTAL-TAE) exam is a certification for experienced software testers who specialize in test automation. ISTQB Certified Tester Advanced Level, Test Automation Engineering certification is designed to validate the skills and knowledge required to design, develop, and maintain automated test systems. The CTAL-TAE Exam is offered by the International Software Testing Qualifications Board (ISTQB) and is recognized worldwide as a standard for advanced-level testing professionals.

ISQI ISTQB Certified Tester Advanced Level, Test Automation Engineering Sample Questions (Q19-Q24):

NEW QUESTION # 19
Assume that you are the TAE responsible for the correct functioning of a TAS, deployed in a test environment that consists of a few machines running the same version of the operating system. The TAS has been working and stable since its deployment, it has been used to run an automated test suite consisting of many similar automated test. The infrastructure team is planning to update the operating system on these machines by installing a new the service pack for security reasons. Since the vendor of the operating system assurance full backward compatibility, the infrastructure team assurance that there will be no impacts on the functioning of the TAS.
What is the BEST approach to confirm the correct functioning of the TAS in this scenario?

- A. Make sure that the infrastructure team has completed installing the service pack on the machines where SUT is running, then run the whole automated test suite to verify itsbehavior
- B. Verify the behavior of the automated tests by running a small tests, then gradually run the remaining tests to confirm the correct functioning of the whole automated test suite.
- C. Do not run any tests because you can immediately confirm the correct functioning of the automated test suite
- D. Verify the behavior of the whole automated test suite by running all the automated tests

Answer: B

NEW QUESTION # 20
A release candidate of a SUT, after being fully integrated with all other necessary systems, has successfully passed all required functional tests (90% were automated tests and 10% were manual tests). Now, it is necessary to perform reliability tests aimed at evaluating whether, under certain conditions, that release will be able to guarantee an MTBF (Mean Time Between Failures) in the production environment higher than a certain threshold (expressed in CPU time). Which of the following test environments is BEST suited to perform these reliability tests?

- A. Integration environment
- B. Local development environment
- C. Preproduction environment
- D. Build environment

Answer: C

Explanation:
Reliability testing (e.g., long-duration runs, endurance/soak, stability measurements, MTBF assessment) requires an environment that closely resembles production in terms of configuration, resource allocation, deployment topology, integrations, and operational characteristics. TAE guidance emphasizes that measurements like MTBF are highly sensitive to environmental differences such as CPU quotas, background load, database sizing, network topology, virtualization settings, and monitoring agents. A local development environment is unsuitable because it is not representative, is often unstable, and typically lacks full system integration. A build environment focuses on building/packaging and fast verification, not production-like reliability evaluation. An integration environment can validate that systems work together, but it is frequently shared, changes often, and may not match production sizing and operational constraints; it is also commonly disrupted by other teams' deployments. Preproduction (often called staging) is designed to be the closest safe approximation to production while still allowing controlled testing, including reliability and performance- related evaluations, without risking real users or live data. Therefore, preproduction is the best-suited environment to run reliability tests intended to predict production MTBF behavior with credible confidence.

NEW QUESTION # 21
A TAS is used to run on a test environment a suite of automated regression tests, written at the UI level, on different releases of a web app: all executions complete successfully, always providing correct results (i.e., producing neither false positives nor false negatives). The tests, all independent of each other, consist of executable test scripts based on the flow model pattern which has been implemented in a three-layer TAF (test scripts, business logic, core libraries) by expanding the page object model via the facade pattern. Currently the suite takes too long to run, and the test scripts are considered too long in terms of LOC (Lines of Code).
Which of the following recommendations would you provide for improving the TAS (assuming it is possible to perform all of them)?

- A. Modify the architecture of the SUT to improve its testability and, if necessary, the TAA accordingly
- B. Implement a mechanism to automatically reboot the entire web app in the event of a crash
- C. Modify the TAF so that test scripts are based on the page object model, rather than the flow model pattern
- D. Split the suite into sub-suites and run each of them concurrently on different test environments

Answer: D

Explanation:
The primary problem is execution time; correctness and independence are already strong. TAE recommends improving feedback time for long-running regression suites by parallelizing execution when tests are independent and the infrastructure supports it. Because the tests are explicitly independent, they are well- suited to parallel execution across multiple environments (or multiple nodes within an environment), reducing overall wall-clock duration without changing test intent. Option B addresses crash recovery, but the scenario says executions complete successfully; crash recovery does not solve the current bottleneck. Option A changes the modeling pattern; it may or may not reduce LOC, but it introduces risk and rework without directly addressing runtime. Also, flow model and facade-expanded page objects are already architectural choices aimed at maintainability and reuse; replacing them is not the most direct solution for speed. Option D (improving SUT testability) can help in general, but it is invasive, expensive, and not targeted to the stated issue when tests already yield correct results. Therefore, the best improvement is to split the suite and run parts concurrently on different environments to reduce total execution time, consistent with TAE guidance on scaling automation execution.

NEW QUESTION # 22
As a TAE, you are evaluating a test automation tool to automate some UI tests for a web app. The automated tests will first locate the required HTML elements on the web page using their corresponding identifiers (locators), then perform actions on those elements, and finally check the presence of any expected text for an HTML element. These tests are independent of each other and are organized into a test suite that must be run every night against the most recent build of the web app. There is a high risk that the web app will crash while running some automated tests. Based only on the given information, which of the following is your MOST important concern related to the evaluation of the test automation tool?

- A. Does the test automation tool offer a feature to create a mock server that simulates the behavior of a real API by accepting requests and returning responses?
- B. Does the test automation tool offer a feature to restore the web app, recover from the failed test, skip such tests, and resume the next one in the suite?
- C. Does the test automation tool support a licensing scheme that allows accessing different feature sets?
- D. Does the test automation tool provide a feature to specify automated tests in a descriptive meta- language that is not directly executable on the web app?

Answer: B

Explanation:
Given the explicit risk that the web app may crash during execution, the highest-priority tool capability is resilience: the ability to recover, continue, and provide usable results from unattended nightly runs. TAE emphasizes that automation must be reliable as a process, not just at the single-test level. If one crash aborts the entire suite, the organization loses feedback for many tests, reduces confidence in the pipeline, and increases triage cost. Therefore, capabilities such as automatic restart of the browser/app, test isolation, robust teardown, failure handling, skipping/marking affected tests, and resuming execution with proper reporting are critical evaluation criteria. Option A (descriptive meta-language) can help readability or non-coder authoring but is not the most urgent need based on the scenario. Option C (mock server) is useful for isolating dependencies in some test levels, but the scenario is UI tests against the most recent build; nothing indicates an API dependency problem that drives tool selection here. Option D (licensing feature sets) affects procurement, but it does not directly mitigate the stated operational risk. Hence, recovery and continuation support is the most important concern.

NEW QUESTION # 23

The GUI of a Customer Relationship Management (CRM) application has been delivered through internet Explorer with proprietary Active X and Java controls. This implementation enables rich client capabilities, but specific commercial automation tools are necessary to automate test cases at GUI of functional test cases.
This is to demonstrate whether a small set of the commercial are able to properly recognize actions taken by a tester when interacting with GUI of the CRM application.
Which of the following scripting techniques would be MOST suitable in this scenario?

- A. Structure scripting
- B. Keyword-driven scripting
- C. Linear scripting
- D. Data-driven scripting

Answer: A

NEW QUESTION # 24
.....