

Familienname, Vorname: _____

Firmenadresse: _____

Telefon: _____

Fax: _____

E-Mail-Adresse: _____

Rechnungsanschrift: _____

Schulungsunternehmen: _____

Referent: _____

**ISTQB® Certified Tester Advanced Level
Test Automation Engineering (CTAL-TAE)
Probeprüfung**

Version v2.2 DE

CTAL-TAE Syllabus v2.2 DE 2025

Urheberrecht

Copyright-Hinweis © International Software Testing Qualifications Board (im Folgenden ISTQB® genannt)

ISTQB® ist eine eingetragene Marke des International Software Testing Qualifications Board.

Copyright © 2024 die Autoren des Lehrplans Test Automation Engineering v2.0: Andrew Pollner (Vorsitz), Péter Földházi, Patrick Quilter, Gergely Ágnesz, László Szikszai

Copyright © 2016 die Autoren Andrew Pollner (Vorsitz), Bryan Bakker, Armin Born, Mark Fewster, Jani Haukinen, Raluca Popescu, Ina Schieferdecker.

Alle Rechte vorbehalten. Die Autoren übertragen hiermit das Urheberrecht an das ISTQB®. Die Autoren (als derzeitige Inhaber der Urheberrechte) und das ISTQB® (als künftiger Inhaber der Urheberrechte) haben sich mit den folgenden Nutzungsbedingungen einverstanden erklärt:

Auszüge aus diesem Dokument dürfen für nicht-kommerzielle Zwecke kopiert werden, wenn die Quelle angegeben wird. Jeder zugelassene Schulungsanbieter darf diesen Lehrplan als Grundlage für einen Schulungskurs verwenden, wenn die Autoren und das ISTQB® als Quelle und Urheberrechtsinhaber des Lehrplans genannt werden und vorausgesetzt, dass in der Werbung für einen solchen Schulungskurs der Lehrplan erst dann erwähnt wird, wenn die offizielle Akkreditierung des Schulungsmaterials durch ein vom ISTQB® anerkanntes Member Board erfolgt ist.

Jede Einzelperson oder Gruppe von Einzelpersonen darf diesen Lehrplan als Grundlage für Artikel und Bücher verwenden, wenn die Autoren und das ISTQB® als Quelle und Urheberrechtsinhaber des Lehrplans genannt werden.

Jede andere Verwendung dieses Lehrplans ist ohne vorherige schriftliche Zustimmung des ISTQB® verboten.

Jedes vom ISTQB® anerkannte Mitgliedskomitee darf diesen Lehrplan übersetzen, sofern es den oben genannten Copyright-Hinweis in der übersetzten Version des Lehrplans wiedergibt.

Danksagung

Dieses Dokument wurde von einem Kernteam des ISTQB® erstellt:

Andrew Pollner (chair), Péter Földházi, Patrick Quilter, Gergely Ágnez, and Geza Bujdoso.

Das Kernteam dankt dem Reviewteam der Arbeitsgruppe Exam, dem Reviewteam der Lehrplanarbeitsgruppe, der Arbeitsgruppe Lehrplan und den Mitgliedsboards für Ihre Vorschläge und Befunde.

Das Technical review wurde durchgeführt von Judy McKay and Gary Mogyorodi.

An der deutschsprachigen Lokalisierung haben mitgearbeitet: Jan Giesen, Marc-Florian Wendland, Horst Pohlmann, Thorsten Geiselhart und Stephan Weissleder.

Änderungsübersicht

Version	Datum	Bemerkungen
v1.0	21.10.2016	CTAL-TAE GA Release
v2.0	03.05.2024	CTAL-TAE v2.0 GA Release
v2.1	04.07.2024	Korrektur der Nummerierung der Fragen 3 und 5
v2.2	16.10.2024	Änderungen der Fragen 1, 2, 3, 5, 9, 10, 11, 13, 16, 18, 19, 20, 23, 24, 25, 26, 28, 29, 30
v.2.2	12.09.2025	Deutschsprachige Lokalisierung auf der Basis der der englischsprachigen Fassung v2.2 mit Änderungen

Einführung

Dies ist eine Probeprüfung. Sie hilft den Kandidaten bei ihrer Vorbereitung auf die Zertifizierungsprüfung. Enthalten sind Fragen, deren Format der regulären ISTQB®/GTB Certified Tester Foundation Level Prüfung ähnelt. Es ist strengstens verboten, diese Prüfungsfragen in einer echten Prüfung zu verwenden.

- 1) Jede Einzelperson und jeder Schulungsanbieter kann diese Probeprüfung in einer Schulung verwenden, wenn ISTQB® als Quelle und Copyright-Inhaber der Probeprüfung anerkannt wird.
- 2) Jede Einzelperson oder Gruppe von Personen kann diese Probeprüfung als Grundlage für Artikel, Bücher oder andere abgeleitete Schriftstücke verwenden, wenn ISTQB® als Quelle und Copyright-Inhaber der Probeprüfung bestätigt wird.
- 3) Jedes vom ISTQB® anerkannte nationale Board kann diese Probeprüfung übersetzen und öffentlich zugänglich machen, wenn ISTQB® als Quelle und Copyright-Inhaber der Probeprüfung bestätigt wird.
- 4) Zu fast jeder Frage wird genau eine zutreffende Lösung erwartet. Bei den Ausnahmen wird explizit auf die Möglichkeit mehrerer Antworten hingewiesen.

Allgemeine Angaben zur Probeprüfung

Anzahl der Fragen: 40

Dauer der Prüfung: 90 Minuten

Gesamtpunktzahl: 66

Punktzahl zum Bestehen der Prüfung: 43 (oder mehr)

Prozentsatz zum Bestehen der Prüfung: 65 % (oder mehr)

Frage 1	TAE-1.1.1	K2	Punkte 1.0
---------	-----------	----	------------

Welche der folgenden Aussagen beschreibt eine Einschränkung der Testautomatisierung?

Wählen Sie EINE Option! (1 aus 4)

a)	Testautomatisierung kann nur für Schnittstellentests wirklich effektiv genutzt werden.	☐
b)	Die Testautomatisierung kann nur auf manuell ausführbare Tests angewendet werden.	☐
c)	Die Testautomatisierung kann nur Ergebnisse überprüfen, die visuell verifiziert werden können.	☐
d)	Die Testautomatisierung kann nur Ergebnisse überprüfen, die automatisiert bzw. maschinell überprüfbar sind.	☒

TAE-1.1.1 (K2) Der Lernende kann die Vor- und Nachteile der Testautomatisierung erklären.

Begründung:

- a) FALSCH – Die Testautomatisierung ist nicht auf Schnittstellentests beschränkt, sondern kann insbesondere da unterstützen, wo manuelle Tests an ihre Grenzen kommen, siehe Lehrplan, Kapitel 1.1.1, zweite Aufzählung, zweiter Spiegelstrich.
- b) FALSCH – Testautomatisierung ist effektiver und effizienter als manuelle Tests, siehe Lehrplan, Kapitel 1.1.1, erste Aufzählung, zweiter Spiegelstrich.
- c) FALSCH – Testautomatisierung beschränkt sich nicht auf visuelle Überprüfungsergebnisse, sondern kann alles prüfen, was mit einem automatisierten Testorakel verglichen werden kann. Siehe Lehrplan, Kapitel 1.1.1, letzte Aufzählung, letzter Spiegelstrich.
- d) **KORREKT** – Wie im Lehrplan steht: „Testautomatisierung kann nur Testergebnisse überprüfen, die von einem automatisierten Testorakel verifiziert werden können.“ Das bedeutet, dass automatisierte Skripte nur solche Ergebnisse kontrollieren können, die maschinell überprüfbar sind. Siehe Lehrplan, Kapitel 1.1.1, letzte Aufzählung, letzter Spiegelstrich.

Frage 2	TAE-1.2.1	K2	Punkte	1.0
---------	-----------	----	--------	-----

Welche der folgenden Aussagen über Testautomatisierung und den Einsatz in den verschiedenen Softwareentwicklungslebenszyklus-Modellen (SDLC) trifft zu?

Wählen Sie EINE Option! (1 aus 4)

a)	In der agilen Softwareentwicklung beschränken sich automatisierte Tests in der Regel auf Akzeptanz- und Komponententests.	☐
b)	Im Wasserfallmodell findet die Implementierung der Testautomatisierung immer während der Verifizierungsphase statt.	☐
c)	Im V-Modell erfolgt die automatisierte Testdurchführung nach der manuellen Testdurchführung.	☐
d)	Im V-Modell wird das Testautomatisierungsframework (TAF) in Abhängigkeit der Bedürfnisse für die jeweilige Teststufe bereitgestellt.	☒

TAE-1.2.1 (K2) Der Lernende kann den Einsatz der Testautomatisierung in verschiedenen Softwareentwicklungslebenszyklus-Modellen erklären.

Begründung:

- a) FALSCH – Laut Lehrplan, Kapitel 1.2.1, Absatz “Agile Softwareentwicklung” können Teams alle Teststufen mit einer Testautomatisierung in angemessenen Umfang und Tiefe abdecken. Es sollte entsprechend auch keine Teststufe kategorisch ausgeschlossen werden.
- b) FALSCH – Die Implementierung der Testautomatisierung erfolgt normalerweise parallel zur oder nach der Implementierungsphase, siehe Lehrplan, Kapitel 1.2.1, Abschnitt “Wasserfall”.
- c) FALSCH – Das V-Modell schreibt keine solche Reihenfolge vor. Es leitet aus den Prozessschritten die traditionellen Teststufen ab, siehe Lehrplan, Kapitel 1.2.1, Abschnitt “V-Modell”. Aber macht keine Vorgaben ob und wann automatisiert werden muss.
- d) **KORREKT** – Die Bereitstellung eines Testautomatisierungsframeworks (TAF) für jede Teststufe ist möglich, siehe Lehrplan, Abschnitt 1.2.1, Absatz “V-Modell”.

Frage 3	TAE-1.2.2	K2	Punkte 1.0
---------	-----------	----	------------

Welcher der folgenden Faktoren muss bei der Auswahl geeigneter Testautomatisierungswerkzeuge NICHT berücksichtigt werden?

Wählen Sie EINE Option! (1 aus 4)

a)	Architektur des SUT	☐
b)	Zusammensetzung und Erfahrung des Testteams	☐
c)	Lizenzmodell und Support des Testwerkzeugs	☐
d)	Anzahl an eingesetzten Testwerkzeugen	☒

TAE-1.2.2 (K2) Der Lernende kann geeignete Testautomatisierungswerkzeuge für ein bestimmtes System unter Test auswählen.

Begründung:

- a) FALSCH – Die Architektur des SUT muss unbedingt beachtet werden, je nach verwendeter Architektur und dadurch bedingten Features können unterschiedliche Werkzeuge zum Einsatz kommen, siehe Lehrplan, Kapitel 1.2.2, Absatz 2 und S.12 im Kapitel 0.11 Aufbau des Lehrplans unter Kapitel 1 steht: "Der Tester lernt, wie sich die Architektur eines SUT auf die Eignung von Testwerkzeugen auswirkt."
- b) FALSCH – Die Zusammensetzung und Erfahrung des Testteams sind entscheidend für den erfolgreichen Einsatz eines Werkzeugs, siehe Lehrplan, Kapitel 1.2.2, Absatz 3.
- c) FALSCH – "Die Verwendung kommerzieller Standardsoftware oder auch die Implementierung einer benutzerdefinierten Lösung auf Basis von Open-Source-Technologie kann durchaus ein komplexer Prozess sein" (Lehrplan, Kapitel 1.2.2, Absatz 2). Daher sollte das bei der Auswahl berücksichtigt werden.
- d) KORREKT – "Es gibt keine Begrenzung für die Anzahl der Testautomatisierungswerkzeuge und -funktionen, die verwendet oder ausgewählt werden können" siehe Lehrplan, Kapitel 1.2.2, Absatz 2.

Frage 4	TAE-2.1.1	K2	Punkte 1.0
---------	-----------	----	------------

Wenn ein System auf Testbarkeit ausgelegt ist, gehört zu den Eigenschaften, dass das Testautomatisierungsframework (TAF) auf Schnittstellen zugreifen kann, um Aktionen auf dem System auszuführen.

Wie heißt diese Eigenschaft?

Wählen Sie EINE Option! (1 aus 4)

a)	Beobachtbarkeit	☐
b)	Steuerbarkeit	☒
c)	Wartbarkeit	☐
d)	Interoperabilität	☐

TAE-2.1.1 (K2) Der Lernende kann Anforderungen an die Konfiguration einer Infrastruktur beschreiben, um Testautomatisierung zu ermöglichen.

Begründung:

- a) FALSCH – Beobachtbarkeit bedeutet, dass man Einblick in den Systemzustand erhält, siehe Lehrplan, Kapitel 2.1.1, 2. Aufzählung, 1. Spiegelstrich.
- b) KORREKT – Steuerbarkeit bezeichnet die Fähigkeit, Aktionen über Schnittstellen zu steuern (siehe Lehrplan, Kapitel 2.1.1, 2. Aufzählung, 2. Spiegelstrich).
- c) FALSCH – Wartbarkeit ist ein Qualitätsmerkmal, dass den Grad bestimmt, zu dem das TAF gewartet werden kann, siehe auch ISTQB Glossar.
- d) FALSCH – Interoperabilität beschreibt die Fähigkeit verschiedener Systeme zur Zusammenarbeit und ist ebenfalls ein Qualitätsmerkmal (siehe auch ISTQB Glossar).

Frage 5	TAE-2.1.2	K2	Punkte 1.0
---------	-----------	----	------------

Welche Art der Testautomatisierung wird hauptsächlich in der Vorproduktionsumgebung eingesetzt?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Komponententest	☐
b)	Benutzerabnahmetests	☒
c)	Statische Analysen	☐
d)	Low-Level-Tests	☐
e)	Nicht-funktionale Tests	☒

TAE-2.1.2 (K2) Der Lernende kann den Einsatz einer Testautomatisierung in verschiedenen Umgebungen erläutern.

Begründung:

- a) FALSCH – Komponententests werden typischerweise in der Build- oder Entwicklungsumgebung ausgeführt (siehe Lehrplan, Kapitel 2.1.2, Absatz “Build-Umgebung”).
- b) KORREKT – Benutzerabnahmetests finden in der Vorproduktionsumgebung statt, um reale Einsatzszenarien unter möglichst produktionsnahen Bedingungen zu prüfen (siehe Lehrplan, Kapitel 2.1.2, Absatz “Vorproduktionsumgebung”).
- c) FALSCH – Statische Analysen werden in der Entwicklungsumgebung durchgeführt (siehe Lehrplan, Kapitel 2.1.2, Absatz “Lokale Entwicklungsumgebung”).
- d) FALSCH – Low-Level-Tests, wie z.B. Komponentenintegrationstests, erfolgen in der Build Umgebung, nicht in der Vorproduktion (siehe Lehrplan, Kapitel 2.1.2, Absatz “Build-Umgebung”).
- e) KORREKT – Nicht-funktionale Tests werden auch insbesondere in Vorproduktionsumgebungen ausgeführt (siehe Lehrplan, Kapitel 2.1.2, Absatz “Vorproduktionsumgebung”).

Frage 6	TAE-2.1.2	K2	Punkte 1.0
---------	-----------	----	------------

Welche Aussage zum Einsatz von Testautomatisierung in verschiedenen Umgebungen ist korrekt?

Wählen Sie EINE Option! (1 aus 4)

a)	In Vorproduktionsumgebungen liegt ein wichtiger Schwerpunkt darin, dass die einzelnen Komponenten einzeln für sich richtig funktionieren.	☐
b)	Die Build-Umgebung ist vor allem dazu da, die Software zu bauen, es werden hier nur statische Analysen aber keine dynamischen Tests ausgeführt.	☐
c)	In der Produktions- bzw. Betriebsumgebung kommt zum ersten Mal die komplette Testsuite mit allen funktionalen Tests zum Einsatz.	☐
d)	In der Integrationsumgebung kommen ausschließlich Black-Box-Tests zum Einsatz.	☒

TAE-2.1.2 (K2) Der Lernende kann den Einsatz einer Testautomatisierung in verschiedenen Umgebungen erläutern.

Begründung:

- a) FALSCH – In Vorproduktions- oder Staging-Umgebungen liegt der Fokus meist nicht auf den Komponenten, sondern vor allem auf nicht-funktionale Qualitätsmerkmale, siehe Lehrplan, Kapitel 2.1.2, Abschnitt “Vorproduktionsumgebung”.
- b) FALSCH – Die Build-Umgebung ist auch für erste, einfache Checks gedacht. Also auch Komponententests (siehe Lehrplan, Kapitel 2.1.2, Abschnitt “Vorproduktionsumgebung”).
- c) FALSCH – Funktionale Testsuiten werden in der Regel zum ersten Mal in der Integrationsumgebung vollständig ausgeführt (siehe Lehrplan, Kapitel 2.1.2, Abschnitt “Integrationsumgebung”).
- d) **KORREKT** – In der Integrationsumgebung gibt es nur Black-Box-Tests (d. h. Systemintegrations- und/oder Abnahmetests), siehe Lehrplan, Kapitel 2.1.2, Abschnitt “Integrationsumgebung”.

Frage 7	TAE-2.2.1	K4	Punkte 3.0
---------	-----------	----	------------

Sie arbeiten für ein IT-Unternehmen, welches Software für ein integriertes auf Android basierendes Multimediasystem für Autos entwickelt.

Die Software besteht aus mehreren miteinander kommunizierenden Komponenten. Die Entwickler verfolgen einen testgetriebenen Entwicklungsansatz (TDD). Nach Abschluss der Entwicklung wird die Software an ein anderes IT-Unternehmen übergeben, welches die Software in die Hardware integriert und als Gesamtpaket an einen Automobilhersteller verkauft. Sie analysieren das System unter Test, um die geeignete Testautomatisierungslösung zu bestimmen.

Welche der folgenden Punkte sollten bei der Erfassung der Testautomatisierungsanforderungen durch das entwickelnde IT-Unternehmen berücksichtigt werden?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Die Teststufe Komponententests soll unterstützt werden um die einzelnen Komponenten automatisiert testen zu können.	☒
b)	Die Testaktivität Testentwurf soll automatisiert werden, damit dem Entwickler die Testfälle für TDD jederzeit zur Verfügung stehen.	☐
c)	Die Testart Fahrzeugtest ist erforderlich, um die Funktionalität der Software in möglichst vielen verschiedenen Fahrzeugtypen sicherzustellen.	☐
d)	Der Entwickler entwirft und automatisiert zunächst seine Testfälle und entwickelt seinen Code so lange inkrementell weiter bis diese erfolgreich durchlaufen.	☒
e)	Die Simulation des AppStores muss verfügbar sein, um die erfolgreiche Integration Ihrer Software in diesen zu verifizieren.	☐

TAE-2.2.1 (K4) Der Lernende kann ein System unter Test analysieren, um die geeignete Testautomatisierungslösung zu bestimmen.

Begründung:

- a) **KORREKT – Komponententests werden bereits vom entwickelnden IT-Unternehmen durchgeführt und müssen daher unterstützt werden (siehe Szenario-Beschreibung).**
- b) FALSCH – Falsch im TDD wird nicht der Entwurf, sondern die Testrealisierung automatisiert und durchgeführt (siehe Szenario und Glossar Testrealisierung, Testentwurf).
- c) FALSCH – Die Auswahl und Durchführung von fahrzeugspezifischen Tests obliegt dem Integrationsunternehmen, da es das Endprodukt an die Kunden verkauft. Die entwickelnde Firma kennt die Zielfahrzeuge gar nicht (siehe Szenario).
- d) **KORREEKT – Dies ist die Vorgehensweise im testgetriebenen Entwicklungsansatz (siehe Szenario und Glossar TDD).**
- e) FALSCH – Ein Test in der simulierten Umgebung eines AppStores hängt von den Zielsystemen ab. Die entwickelnde Firma kennt weder die Zielsysteme noch die Randbedingungen und kann daher keine Tests gegen eine Simulation durchführen. Dies ist ebenfalls Aufgabe der integrierenden Firma (siehe Szenario).

Frage 8	TAE-2.2.2	K4	Punkte 3.0
---------	-----------	----	------------

Sie bewerten Testautomatisierungswerkzeuge. Für eines der Werkzeuge liegen folgende Erkenntnisse vor:

- Ein sehr informatives Dashboard zeigt alle relevanten Testinformationen zum SUT an.
- Eine Testprotokollierungskomponente erfasst alle notwendigen Informationen zur Fehlerbehebung bei auftretenden Problemen.
- Eine anpassbare Testberichterstattungskomponente ist vorhanden.
- Im Proof of Concept arbeitete das Werkzeug im Vergleich zu den anderen getesteten Tools sehr langsam.
- Die aktuelle Testumgebung erfüllt laut Release Notes alle Hardware und Softwareanforderungen.

Welche der folgenden Antworten sollte Ihr nächster Schritt bei der Auswahl dieses Werkzeugs sein?

Wählen Sie EINE Option! (1 aus 4)

a)	Deutlich mehr Hardware Ressourcen für die Live-Umgebung einplanen, um Leistungseinbußen auszuschließen.	☐
b)	Die Testprotokollierung vollständig deaktivieren, um die Performance des Werkzeugs zu verbessern.	☐
c)	Aufgrund der Performance von der Auswahl dieses Werkzeugs abraten.	☒
d)	Das SUT auf die neueste Version aktualisieren, damit sich die Performance des Werkzeuges steigert.	☐

TAE-2.2.2 (K4) Der Lernende kann technische Befunde einer Werkzeugevaluation veranschaulichen.

Begründung:

- a) FALSCH – Die Testumgebung erfüllt bereits alle in den Release Notes genannten Anforderungen (siehe Szenario).
- b) FALSCH – Die Protokollierung ist für die Fehlersuche unerlässlich und darf nicht vollständig deaktiviert werden (siehe Szenario).
- c) KORREKT – Da das Werkzeug bereits in einer Testumgebung eine schlechte Performance liefert, sollte es nicht weiterverfolgt werden, da in der Live-Umgebung nicht mit besserer Performance zu rechnen ist – im Gegenteil wird das Tool im Live-Betrieb aufgrund intensiverer Nutzung höchstwahrscheinlich noch langsamer sein (siehe Szenario).
- d) FALSCH – Die Version des SUT sollte keinen Einfluss auf die Performance des Werkzeuges haben (siehe Szenario).

Frage 9	TAE-3.1.1	K2	Punkte 1.0
---------	-----------	----	------------

Ordnen Sie die folgenden Aufgaben den entsprechenden Hauptfunktionen der Testautomatisierungsarchitektur zu:

1. Automatisierte Ableitung von Testfällen aus Testmodellen.
2. Erstellung von Schlüsselwort-Testfällen unter Verwendung von Schlüsselwort-Testbibliotheken.
3. Testprotokollierung mit detaillierten Informationen über Testschritte und SUT-Verhalten.
4. Anbindung einer automatisierten Testsuite über einen MQTT-Adapter an das SUT.

Hauptfunktionen:

- A. Testdefinition
- B. Testanpassung
- C. Testgenerierung
- D. Testausführung

Wählen Sie EINE Option! (1 aus 4)

a)	1 → D, 2 → C, 3 → B, 4 → C	☐
b)	1 → C, 2 → A, 3 → B, 4 → D	☐
c)	1 → A, 2 → B, 3 → D, 4 → C	☐
d)	1 → C, 2 → A, 3 → D, 4 → B	☒

TAE-3.1.1 (K2) Der Lernende kann die wichtigsten Funktionen einer Testautomatisierungsarchitektur erläutern.

Begründung:

- a) 1 → C (Testgenerierung). Diese Hauptfunktion unterstützt den automatisierten Entwurf von Testfällen (z. B. modellbasierte Testfallgenerierung). (Siehe Syllabus, Kapitel 3.1.1 Absatz Testgenerierung).
- b) 2 → A (Testdefinition). Hier erfolgt die konkrete Implementierung der Testfälle und Testsuiten (siehe Syllabus, Kapitel 3.1.1 Absatz Testdefinition) sowie die Verwaltung von Testbibliotheken.
- c) 3 → D (Testdurchführung). Diese Fähigkeit umfasst die Ausführung der Tests sowie das Protokollieren der Testausführung (siehe Syllabus, Kapitel 3.1.1 Absatz Testdurchführung).
- d) 4 → B (Testanpassung). Adapter und Verbindungsmechanismen passen automatisierte Tests an die Schnittstellen des SUT an (siehe Syllabus, Kapitel 3.1.1 Absatz Testanpassung).

Damit ist Antwort d) korrekt.

Frage 10	TAE-3.1.2	K2	Punkte 1.0
----------	-----------	----	------------

Welche der folgenden Aspekte müssen NICHT ZWINGEND im technischen Entwurf der Testautomatisierungslösung (TAS) enthalten sein?

Wählen Sie EINE Option! (1 aus 4)

a)	Auswahl der Werkzeuge für die Testautomatisierung und der werkzeugspezifischen Bibliotheken.	☐
b)	Analyse und Identifikation der API-Anforderungen für das Werkzeug der Testautomatisierung.	☐
c)	SUT-spezifische Anpassungen für das ausgewählte Werkzeug für die Testautomatisierung.	☒
d)	Konnektoren zwischen der TAS und dem Testmanagementwerkzeug.	☐

TAE-3.1.2 (K2) Der Lernende kann erläutern, wie man eine Testautomatisierungslösung entwirft.

Begründung:

- a) FALSCH – Eine Auswahl der werkzeugspezifischen Bibliotheken ist zu treffen. Siehe TAE-Lehrplan v2.0, Kapitel 3.1.2, 2. Absatz, Aufzählungspunkt Werkzeuge: „Auswahl von Werkzeugen zur Testautomatisierung und werkzeugspezifischen Bibliotheken“.
- b) FALSCH – Die Anforderungen an die Konnektivität/Schnittstellen (z. B. API) des Werkzeugs für die Testautomatisierung soll im techn. Entwurf definiert sein. Siehe TAE-Lehrplan v2.0, Kapitel 3.1.2, 2. Absatz, Aufzählungspunkt Schnittstellen: „Identifizierung von Anforderungen an Konnektivität und Schnittstellen (z. B. Firewalls, Datenbanken, URLs/Verbindungen, Mocks/Platzhalter, Message Queues und Protokolle)“.
- c) KORREKT – Die TAS benötigt NICHT zwingend SUT-spezifische Anpassungen. Standardisierte Werkzeuge für Testautomatisierung sind mit vielen der verbreiteten Schnittstellen kompatibel. Siehe Syllabus, Kapitel 3.1.2 1. Absatz, zweiter Satz. „Eine TAS ... benötigt möglicherweise zusätzliche SUT-spezifische Anpassungen“.
- d) FALSCH – Eine Verbindung zwischen Testmanagement und TAS ist erforderlich, um die automatisierten Testfälle in der TAS mit den Definitionen im Testmanagement verbinden zu können. Siehe Syllabus, Kapitel 3.1.2 zweiter Absatz Aufzählungspunkt Testmanagement sowie Kapitel 3.1.1 Absatz Schnittstellen.

Frage 11	TAE-3.1.3	K3	Punkte 2.0
----------	-----------	----	------------

Sie pflegen ein Testautomatisierungsframework (TAF) für Android-Applikationen. Nun soll die dazugehörige Testautomatisierungslösung (TAS) auch iOS-Applikationen unterstützen. Da Zeit und Budget knapp sind, wurden Sie gebeten, das bestehende TAF an die neuen Anforderungen anzupassen.

In welcher Schicht konfigurieren Sie die Verbindung zur neuen Anwendung?

Wählen Sie EINE Option! (1 aus 4)

a)	Schicht der Kernbibliotheken	☐
b)	Schicht der Testskripte	☐
c)	Schicht der Testausführung	☐
d)	Schicht der Geschäftslogik	☒

TAE-3.1.3 (K3) Der Lernende kann die Schichten von Testautomatisierungsframeworks anwenden.

Begründung:

- a) FALSCH – Die Kernbibliotheken sind anwendungsunabhängig und sollten daher keine SUT-spezifischen Logiken oder Informationen enthalten. Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.3, Abschnitt Kernbibliotheken.
- b) FALSCH – Testskripte sollten ausschließlich Testabläufe beschreiben, aber keine Details über Verbindung zum SUT beinhalten. Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.3, Abschnitt Testskripte.
- c) FALSCH – Schicht der Testausführung gehört nicht zu den TAF-Schichten. Diese Schicht gehört zur TAS. Abgesehen davon, ist die Testausführungsschicht nicht für die Verbindung zum SUT gedacht. Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.1, Abschnitt Fähigkeiten von Werkzeugen und Bibliotheken zur Testautomatisierung.
- d) **KORREKT – In der Geschäftslogik-Schicht wird u.a. die Verbindung zum SUT eingerichtet und konfiguriert. Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.3, Abschnitt Geschäftslogik.**

Frage 12	TAE-3.1.4	K3	Punkte 2.0
----------	-----------	----	------------

Sie arbeiten an einem Testautomatisierungsprojekt, das GUI-Tests für einen webbasierten öffentlichen Verkehrsdienst automatisiert. Das Projekt hat einen engen Zeitrahmen. Es liegen manuelle Testfälle vor, die möglichst schnell automatisiert werden sollen. Ein Ziel ist, Testfälle direkt in automatisierte Testskripte zu überführen.

Welche beiden Ansätze sollten AM EHESTEN eingesetzt werden, um dieses Ziel zu erreichen?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Schlüsselwortgetriebenes Testen	☐
b)	Verhaltensgetriebene Entwicklung (BDD)	☐
c)	Mitschnitt	☒
d)	Datengetriebenes Testen	☐
e)	Lineare Skripterstellung	☒

TAE-3.1.4 (K3) Der Lernende kann die verschiedenen Ansätze zur Automatisierung von Testfällen anwenden.

Begründung:

- a) FALSCH – Schlüsselwortgetriebenes Testen erfordert aufwändige Framework-Erweiterungen und ist für schnelle Automatisierung nicht optimal, siehe Syllabus, Kapitel 3.1.4 Absatz „Schlüsselwortgetriebenes Testen“.
- b) FALSCH – BDD bringt einen hohen initialen Aufwand für Feature-Definitionen und ist in einem engen Zeitrahmen nicht praktikabel, siehe Syllabus, Kapitel 3.1.4 Absatz „Verhaltensgetriebene Entwicklung“.
- c) KORREKT – Der Mitschnitt lässt sich (vor allem anfänglich) sehr schnell aufsetzen und verwenden, daher eignet er sich gut für die zügige Automatisierung vorhandener Testfälle, siehe Syllabus, Kapitel 3.1.4 Absatz „Mitschnitt“.
- d) FALSCH – Datengetriebenes Testen erfordert umfangreiche Testdaten-Frameworks und sind für eine kurzfristige Lösung zu komplex, siehe Syllabus, Kapitel 3.1.4 Absatz „Datengetriebenes Testen“.
- e) KORREKT – Die lineare Skripterstellung ist einfach einzurichten und umzusetzen. Insbesondere im Zusammenspiel mit dem Mitschnittansatz liefert die lineare Skripterstellung schnelle und brauchbare Ergebnisse in Bezug auf das Szenario. Wie der Mitschnittansatz eignet sie sich daher gut für die zügige Automatisierung vorhandener Testfälle, siehe Syllabus, Kapitel 3.1.4 Absatz „Lineare Skripterstellung“.

Frage 13	TAE-3.1.4	K3	Punkte 2.0
----------	-----------	----	------------

Ordnen Sie die Fragmente von Testskripten, die unterschiedliche Testautomatisierungsansätze repräsentieren, den jeweiligen Testautomatisierungsansätzen zu:

Testfälle (bzw. Fragmente davon):

A. testfall create_user (\$Benutzername, \$Passwort, \$mail)

```
{...
```

```
testbibliothek.setMailAddress($mail)
```

```
....}
```

B. testfall create_user()

```
{...
```

```
    var user = new User("Max Mustermann", "GeheimesPasswort",  
    max@mustermann.example);
```

```
...}
```

C. Create User \$Benutzername \$Passwort \$Mail

D. Start Sequenzaufzeichnung:

Textfeld "Benutzername" eingeben: Max Mustermann

Textfeld "E-Mail" eingeben: max@mustermann.example

Textfeld "Passwort" eingeben: geheimesPasswort

Button "Create Account" klicken

Testautomatisierungsansätze:

1. Lineare Skripterstellung
2. Mitschnittansatz
3. Strukturierte Skripterstellung
4. Schlüsselwortgetriebenes Testen

Wählen Sie EINE Option! (1 aus 4)

a)	1-A, 2-B, 3-C, 4-D	☐
b)	1-B, 2-D, 3-A, 4-C	☒
c)	1-B, 2-A, 3-D, 4-C	☐
d)	1-C, 2-A, 3-D, 4-B	☐

TAE-3.1.4 (K3) Der Lernende kann die verschiedenen Ansätze zur Automatisierung von Testfällen anwenden.

Begründung:

Die richtige Kombination ist 1-B, 2-D, 3-A, 4-C und damit Option b)

Die folgenden Kombinationen sind korrekt:

Kombination 1-B: Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.4, Unterkapitel Lineare Skripterstellung: „Die lineare Skripterstellung ist eine Programmierfähigkeit, die keine von TAEs erstellten Testbibliotheken“ Das vorliegende Fragment verwendet keinerlei Testbibliotheken, noch folgt es einem datengetriebenen Ansatz; folglich basiert dieser Testfall auf der linearen Skripterstellung.

Kombination 2-D: Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.4, Unterkapitel Mitschnittansatz: „Der Mitschnitt (engl. Capture/Playback) ist ein Ansatz, der die Interaktionen mit dem SUT aufzeichnet, während eine Sequenz von Aktionen manuell ausgeführt wird. Diese Werkzeuge erzeugen während der Erfassung Testskripte, und je nach verwendetem Werkzeug kann der Code für die Testautomatisierung modifizierbar sein.“ Der Testfall entspricht dem Mitschnittansatz; die Aktionen nach der Sequenzaufzeichnung entsprechen den manuell aufgezeichneten Schritten.

Kombination 3-A: Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.4, Unterkapitel Strukturierte Skripterstellung: „Es werden Testbibliotheken mit wiederverwendbaren Elementen, Testschritten und/oder User Journeys eingeführt. Für die Erstellung und Wartung von Testskripten sind bei diesem Ansatz Programmierkenntnisse erforderlich.“ Der Testfall ruft u.a. einen Testschritt aus einer Testbibliothek zum Setzen des Benutzernamens auf. Der Testschritt `testbibliothek.setUserName()` kann aus beliebig vielen Testskripten bei Bedarf aufgerufen werden.

Kombination 4-C: Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.4, Unterkapitel Schlüsselortgetriebenes Testen: „Bei schlüsselwortgetriebenen Tests (engl. Keyword Driven Testing, KDT) handelt es sich um eine Liste oder Tabelle von Testschritten, die aus Schlüsselwörtern und den Testdaten abgeleitet werden, auf denen die Schlüsselwörter operieren. Die Schlüsselwörter werden aus der Sicht des Benutzers definiert.“ Create User ist ein solches Schlüsselwort, welches aus Benutzersicht eine fachliche Aktion beschreibt.

Frage 14	TAE-3.1.5	K3	Punkte 2.0
----------	-----------	----	------------

Sie sind an einem Testautomatisierungsprojekt beteiligt, das GUI-Tests für eine E-Commerce-Website automatisiert. Die Webseite enthält u.a. einen neuartigen digitalen Assistenten, der das Anlegen von Benutzerkonten mit Namen, Rechnungs- und Lieferadresse sowie Sicherheitsdaten vereinfacht. Die Entwicklung verläuft iterativ, insbesondere das GUI-Layout ist beinahe in jeder Iteration Gegenstand von Änderungen. Der Workflow des Assistenten hingegen ist weitestgehend stabil. Aktuell müssen die Änderungen an der GUI in jedem Testskript nachgezogen werden, was einen enormen Wartungsaufwand bedeutet. Sie sind gebeten worden kurzfristig für Abhilfe zu sorgen.

Welches Entwurfsmuster empfehlen Sie in der aktuellen Situation, um kurzfristig die Wartungsaufwände zu reduzieren?

Wählen Sie EINE Option! (1 aus 4)

a)	Page-Object-Model: Durch die Trennung der Testskripte von und den UI-Details, müssen Änderungen nur in den Page-Objects angepasst werden und sind dann für alle Testfälle aktuell.	☒
b)	Flow-Model-Pattern: Durch die Abbildung der Workflows des Assistenten, werden die Testskripte weniger anfällig für fachliche Änderungen im Workflow.	☐
c)	Polymorphismus: Durch die Erstellung abstrakter Klassen und deren Überladung in Unterklassen, wird es möglich, die Testskripte an jedes GUI-Layout anzupassen.	☐
d)	Singleton-Muster: Nutzung eines einzigen Objekts für die Elementlokalisierung.	☐

TAE-3.1.5 (K3) Der Lernende kann die Entwurfsprinzipien und Entwurfsmuster in der Testautomatisierung anwenden.

Begründung:

- a) **KORREKT** – Das Page-Object-Model ist in dieser Situation das am besten geeignete Entwurfsmuster. Es konzentriert sich auf die Trennung von Testskripten und den GUI-Elementen zwecks besserer Wartbarkeit der Testskripte. Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.5, Abschnitt Entwurfsmuster, 3. Absatz: “Wann immer sich die Struktur des SUT ändert, muss der TAE nur an einer Stelle Aktualisierungen vornehmen, nämlich am Locator innerhalb eines Seitenmodells, anstatt die Locatoren in jedem Testfall zu aktualisieren. Das Page-Object-Model ist im Gegensatz zum Flow-Model-Pattern einfacher und schneller zu realisieren, da keinerlei Benutzerinteraktionen in dem Model umzusetzen sind.
- b) **FALSCH** – Das Flow-Model-Pattern würde langfristig gesehen sicherlich ebenfalls eine positive Auswirkung auf die Wartbarkeit der Testskripte haben, allerdings ist in diesem Szenario klar kommuniziert, dass der Workflow des Assistenten stabil bleibt. Der Aufwand für die Entwicklung eines Flow-Model-Patterns ist größer als für das Page-Object-Model, weswegen es nicht kurzfristig umsetzbar ist, und ist daher nicht die beste Lösung.
- c) **FALSCH** – Polymorphismus ist kein Entwurfsmuster im eigentlichen Sinne, sondern ein objektorientiertes Grundkonzept. Polymorphismus hat prinzipiell erst einmal keinen positiven Effekt auf die Wartungsaufwände. Zudem ist der beschriebene angedachte Verwendungszweck nur schwer vorstellbar. Vgl. TAE-Lehrplan v2.0, Kapitel 3.1.5 Abschnitt Objektorientierte Programmierprinzipien“.
- d) **FALSCH** – Das Singleton-Muster ist dafür gedacht, um Kontrolle über die Instanziierung von Objekten zu behalten, die Abstraktion bzw. Kapselung von Lokatoren ist nicht das Ziel des Singleton-Musters.

Frage 15	TAE-4.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Die Geschäftsleitung möchte in Ihrer Organisation eine Testautomatisierungslösung (TAS) einführen und hat Sie beauftragt, ein Pilotprojekt zu starten und zu leiten.

Welche der folgenden Punkte können als Grundlage für Anforderungen an dieses Pilotprojekt AM BESTEN genutzt werden?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Identifizierung von Wissen und Erfahrung der einzelnen Teammitglieder.	☒
b)	Identifizierung von Metriken und Messmethoden zur Überwachung des SUT in der Produktionsumgebung.	☐
c)	Bewertung des Effektes der verschiedenen Ansätze.	☐
d)	Evaluierung von Lizenzierungsoptionen und unternehmensspezifischen Vorgaben.	☒
e)	Festlegung des am besten geeigneten CI/CD-Setups.	☐

TAE-4.1.1 (K3) Der Lernende kann Richtlinien anwenden, die eine effektive Pilotierung und Implementierung der Testautomatisierung unterstützen.

Begründung:

- a) **KORREKT** – Die Erfahrung der Teammitglieder sollte berücksichtigt werden (siehe Lehrplan, Kapitel 4.1.1, Aufzählung zu nicht-technischen Aspekten).
- b) **FALSCH** – Die Überwachung des SUT in der Produktion gehört nicht zum typischen Fokus eines Pilotprojekts zur Einführung von TAS.
- c) **FALSCH** – Die Bewertung des Effektes (also des Erfolgs oder der Fehlerwirkung) erfolgt erst nach Abschluss des Pilotprojektes (siehe Syllabus, Kapitel 4.1.1, letzter Absatz).
- d) **KORREKT** – Details zu den Lizenzierungsoptionen und Unternehmensregeln sollten sich auch in den Anforderungen an das Pilotprojekt wieder finden, siehe Lehrplan, Kapitel 4.1.1. Aufzählungen zu nicht-technischen Aspekte.
- e) Das anfängliche CI/CD-Setup der Pilotierung lässt sich jederzeit ändern, siehe Lehrplan, Kapitel 4.1.1, 6. Absatz.

Frage 16	TAE-4.2.1	K4	Punkte 3.0
----------	-----------	----	------------

Ihr Unternehmen entwickelt verteilte Systeme und testet diese auf verschiedenen Teststufen. Nach der Integration werden die Security-Konzepte aktiviert, welche den Zugriff auf bestimmte Schnittstellen unterbinden. Im Systemtest, der in der Produktivumgebung durchgeführt wird, soll trotzdem über diese Schnittstellen automatisiert getestet werden. Entsprechend wird die Konfiguration des Systems für die automatisierten Systemtests angepasst. Die Testsuiten werden für die unterschiedlichen Teststufen aus verschiedenen Repositorien abgerufen. Für jede Teststufe wird eine separate Umgebung für die Testautomatisierung genutzt.

Welches der folgenden Risiken sehen Sie als gegeben?

Wählen Sie EINE Option! (1 aus 4)

a)	Durch die sequenzielle Durchführung von Tests auf verschiedenen Teststufen kann es zu einem Performanzproblem kommen.	☐
b)	Die verschiedenen Ebenen der Testprotokollierung können pro Teststufe unterschiedliche Bedeutungen haben.	☐
c)	Nach dem Systemtest könnten nicht geschlossene Firewall-Ports Zugriff auf einzelne SUT-Schnittstellen weiterhin ermöglichen.	☒
d)	Die Testrahmen müssen häufiger aktualisiert werden, damit unterschiedliche Teststufen problemlos hintereinander durchgeführt werden können.	☐

TAE-4.2.1 (K4) Der Lernende kann Risiken bei der Einführung analysieren und Strategien zur Risikominderung für die Testautomatisierung planen.

Begründung:

- a) FALSCH – Es sind keine Performanzprobleme zu erwarten. Die Tests werden sequenziell durchgeführt und für jede Teststufe gibt es eine separate Umgebung für die Testautomatisierung. Siehe Text und Beschreibung Option a).
- b) FALSCH – Davon ist nicht auszugehen, da die verschiedenen Ebenen eine klare Definition haben und allein aus der Verwendung für verschiedene Teststufen keine Verwirrung zu erwarten ist. Siehe 4.2.1, Abschnitt „Protokollierung“.
- c) **KORREKT – Für die Automatisierung des Systemtests wurden einige Schnittstellen für die Zugriffe „von außen“ geöffnet. Wenn diese Konfiguration nach der Testdurchführung nicht wieder zurückgenommen wird, besteht das Risiko, dass Zugriff bspw. über Firewall-Ports zu eigentlich geschützten Bereichen besteht. Generell ist das Risiko von offenen Firewall-Ports nach dem Test ein generelles Risiko, welches sorgfältig adressiert werden sollte, das nicht unbedingt nur mit der Testautomatisierung zusammenhängt. Siehe 4.2.1, Absatz 3 und 4.**
- d) FALSCH – Die verwendeten Testrahmen sollten für die Testdurchführung klar definiert werden. Allein die Durchführung von Tests auf verschiedenen Stufen spricht nicht für den Bedarf, die Testrahmen häufiger zu aktualisieren – das hängt eher an der Verfügbarkeit von Updates.

Frage 17	TAE-4.3.1	K2	Punkte 1.0
----------	-----------	----	------------

Welche zwei der beiden folgenden Punkte ist ein wichtiger Faktor zur Verbesserung der Wartbarkeit des Codes für die Testautomatisierung?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Definieren von generischen Funktionen mit allen notwendigen Parametern.	☐
b)	Vermeiden Sie typische Entwurfsmuster.	☐
c)	Verwenden Sie statische Analysatoren, um den Code sauber zu halten.	☒
d)	Nutzen Sie hardcodierte Werte, um ihre Bedeutung leicht verständlich zu machen.	☐
e)	Fokussieren Sie sich darauf, den Code testbar zu halten.	☒

TAE-4.3.1 (K2) Der Lernende kann erläutern, welche Faktoren die Wartbarkeit einer Testautomatisierungslösung unterstützen und beeinflussen.

Begründung:

- a) FALSCH – Nur generische Funktionen mit allen Parametern zu definieren, verbessert nicht automatisch die Wartbarkeit (siehe Lehrplan, Aufzählung in Kapitel 4.3.1, Spiegelstriche 4 und 5).
- b) FALSCH – Beliebige, individuelle Variablennamen führen eher zu Inkonsistenzen und erschweren die Wartung (siehe Lehrplan, Aufzählung in Kapitel 4.3.1, Spiegelstrich 7).
- c) KORREKT – Statische Analysatoren erkennen Codesmells, Stilverstöße und potenzielle Fehler frühzeitig und fördern so die Wartbarkeit (siehe Lehrplan, Aufzählung in Kapitel 4.3.1, vorletzter Absatz).
- d) FALSCH – Hartcodierte Werte verringern die Flexibilität und Verständlichkeit des Codes und verschlechtern somit die Wartbarkeit (siehe Lehrplan, Aufzählung in Kapitel 4.3.1, dritter Absatz).
- e) KORREKT – Sie sollten sich auf Testbarkeit der Features fokussieren. Siehe Lehrplan, Aufzählung in Kapitel 4.3.1, letzter Spiegelstrich.

Frage18	TAE-4.3.1	K2	Punkte 1.0
---------	-----------	----	------------

Bringen Sie die folgenden Empfehlungen für eine bessere Wartbarkeit mit den entsprechenden Begründungen zusammen!

1. Hardcoding von Testdaten in einem Testskript sollte vermieden werden, weil ...
 2. Schlüsselwörter sollten nur eine minimale nötige Anzahl von Parametern definieren, weil ...
 3. Entwurfsmuster sollten zielführend verwendet werden, weil ...
 4. Namenskonventionen und Programmierrichtlinien bei der Implementierung von Testskripten verwendet und durch IDEs unterstützt werden, weil ...
- A. ... dadurch die Lesbarkeit und Modifizierbarkeit der Testskripte verbessert wird.
- B. ... dadurch Redundanzen erzeugt werden, die den Wartungsaufwand je Testskript erhöhen.
- C. ... dadurch das Verständnis und die Verwendung innerhalb von Testskripten erleichtert wird.
- D. ... dadurch gut strukturierter und wiederverwendbarer Testcode erzeugt wird.

Wählen Sie EINE Option! (1 aus 4)

a)	1-A, 2-D, 3-C, 4-B	☐
b)	1-B, 2-C, 3-D, 4-A	☒
c)	1-B, 2-C, 3-A, 4-D	☐
d)	1-B, 2-A, 3-D, 4-C	☐

TAE-4.3.1 (K2) Der Lernende kann erläutern, welche Faktoren die Wartbarkeit einer Testautomatisierungslösung unterstützen und beeinflussen.

Begründung:

Die folgende Kombination ist korrekt: 1-B, 2-C, 3-D, 4-A und damit Option b)

Begründung für 1-B „Hardcoding von Testdaten in einem Testskript sollte vermieden werden, weil dadurch Redundanzen erzeugt werden, die den Wartungsaufwand je Testskript erhöhen.“: Hartkodierte Testdaten in Testskripten führen dazu, dass auch Testskripte dupliziert werden müssen, die durch datengetriebenes Testen eigentlich vereinheitlicht werden könnten. Dadurch entstehen zu großen Teilen redundante Testfälle, die sich nur in den hartkodierten Testdaten unterscheiden. Dies erhöht mitunter enorm den Wartungsaufwand. Vgl. TAE-Lehrplan v2.0, Kapitel 4.3.1: „Es kann durch datengetriebene Tests vermieden werden, so dass die Testdaten aus einer gemeinsamen Quelle stammen, die leichter zu pflegen ist. Hardcoding verkürzt die Entwicklungszeit, ist aber nicht empfehlenswert, da sich die Daten häufig ändern können und die Pflege zeitaufwendig ist.“

Begründung für 2-C „Schlüsselwörter sollten nur eine minimale nötige Anzahl von Parametern definieren, weil dadurch das Verständnis und die Verwendung innerhalb von Testskripten erleichtert wird“: Besitzt ein Schlüsselwort sehr viele, unter Umständen sogar optionale Parameter, erschwert dies zum einen die Dokumentation des Schlüsselworts an sich, als auch dessen Einsatz. Die TAEs müssen alle Parameter verstehen und sicher sein, dass das Auslassen eines Parameters bei der Verwendung des Schlüsselworts keinen unerwünschten Seiteneffekt hat. Zudem ist es auch zeitaufwändiger, eine Schlüsselwortimplementierung mit vielen Parametern zu warten.

Begründung für 3-D „Entwurfsmuster sollten zielführend verwendet werden, weil dadurch gut strukturierter und wiederverwendbarer Testcode erzeugt wird.“: Werden Entwurfsmuster zielführend und korrekt eingesetzt, helfen sie, um wiederverwendbare und lesbare Codestrukturen (wie bspw. beim Page-Object-Modell) zu erzeugen. Vgl. TAE-Lehrplan v2.0, Kapitel 4.3.1: „Die Nutzung von Entwurfsmustern ist ebenfalls sehr zu empfehlen. Die Entwurfsmuster - wie in Kapitel 3.1.5 beschrieben - ermöglichen die Implementierung eines strukturierten und gut wartbaren Codes für die Testautomatisierung, sofern die Entwurfsmuster richtig verwendet werden.“

Begründung für 4-A „Namenskonventionen und Programmierrichtlinien sollten durch IDEs unterstützt werden, weil dadurch die Lesbarkeit und Modifizierbarkeit der Testskripte verbessert wird.“: Namenskonventionen und Programmierrichtlinien vereinheitlichen die Art und Weise, wie Strukturen und Konzepte (bspw. Variablen oder Schlüsselwörter) in Testskripten bezeichnet bzw. verwendet werden. Nach einer Einarbeitungszeit lässt sich dadurch schnell die Bedeutung, die hinter einem Konzept in einem Testskript (oder dem Testskript selbst) steckt anhand seines Namens erschließen. Vgl. TAE-Lehrplan v2.0, Kapitel 4.1.3: „Namenskonventionen sind sehr hilfreich, um den Zweck einer bestimmten Variablen zu identifizieren. Verständliche Variablennamen wie „loginButton“, „resetPasswordButton“ helfen den TAEs zu verstehen, welche Komponente zu verwenden ist.“

Frage 19	TAE-5.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Ein Entwicklungsteam implementiert eine CI/CD-Pipeline für ein neues Softwareprojekt.

Wie werden Teststufen AM EHESTEN in die Pipelines integriert?

Wählen Sie EINE Option! (1 aus 4)

a)	Komponententests werden nach der Integrationspipeline ausgeführt, Systemtests werden vor dem Code-Commit ausgeführt, und Abnahmetests werden während der Deployment-Pipeline ausgeführt.	☐
b)	Alle Komponententests und Systemtests werden nach dem Deployment ausgeführt, während Abnahmetests später in der Produktionsumgebung ausgeführt werden.	☐
c)	Komponententests werden während des Builds ausgeführt, Systemtests werden nach dem Deployment in einer Testumgebung ausgeführt, und Abnahmetests werden in einer Vorproduktionsumgebung ausgeführt.	☒
d)	Alle vorhandenen Tests werden gleichzeitig nach dem Deployment in der Produktionsumgebung ausgeführt, um die Durchlaufzeit der Pipeline zu minimieren.	☐

TAE-5.1.1 (K3) Der Lernende kann Testautomatisierung auf verschiedenen Teststufen innerhalb von Pipelines anwenden.

Begründung:

- a) FALSCH – Laut Lehrplan, Kapitel 5.1.1, Aufzählung im 2. Absatz, werden Komponententests typischerweise während des Builds oder vor dem Deployment ausgeführt und nicht nach der Integrationspipeline. Systemtests sollten nicht vor dem Code-Commit ausgeführt werden, da sie in der Regel eine vollständige Systemumgebung benötigen.
- b) FALSCH – Obwohl Komponententests vor dem Deployment ausgeführt werden können, ist es laut Lehrplan, Kapitel 5.1.1, Aufzählung im 2. Absatz nicht die Regel. Abnahmetests werden in der Regel in einer Vorproduktionsumgebung ausgeführt, die der Produktionsumgebung so ähnlich wie möglich ist, siehe auch Lehrplan, Kapitel 2.1.2, Abschnitt “Vorproduktionsumgebung”.
- c) KORREKT – Siehe Lehrplan, Kapitel 5.1.1, Aufzählung im 2. Absatz, beschreibt, dass Komponententests in der Regel während des Builds ausgeführt werden, Systemtests oft Teil einer kontinuierlichen Delivery-Pipeline als Quality Gates sind und nach dem Deployment in einer Testumgebung ausgeführt werden können, und Abnahmetests in der Regel in einer Vorproduktionsumgebung ausgeführt werden, die der Produktionsumgebung so ähnlich wie möglich ist, siehe auch Lehrplan, Kapitel 2.1.2, Abschnitt “Vorproduktionsumgebung”.
- d) FALSCH – Laut Lehrplan, Kapitel 5.1.1, Aufzählung im 2. Absatz, wird nicht empfohlen alle Tests gleichzeitig und erst nach dem Deployment in der Produktionsumgebung auszuführen, sondern die einzelnen Teststufen sollten möglichst in unterschiedlichen Pipelines in unterschiedlichen Entwicklungsschritten aufgeführt sein. Außerdem würde so ein Vorgehen dem "Shift-Left"-Ansatz widersprechen, bei dem Fehler so früh wie möglich im Entwicklungsprozess erkannt werden sollen.

Frage 20	TAE-5.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Welcher der folgenden Ansätze ist am besten geeignet, um Systemintegrationstests in eine CI/CD-Pipeline zu integrieren, wenn die Tests als Quality Gates dienen sollen?

Wählen Sie EINE Option! (1 aus 4)

a)	Die Tests werden in einer separaten Pipeline ausgeführt, die durch ein erfolgreiches Deployment ausgelöst wird.	☐
b)	Die Tests werden als Teil des Deployments nach der Bereitstellung der Komponente ausgeführt.	☒
c)	Die Tests werden in der lokalen Entwicklungsumgebung ausgeführt, bevor Code in das Repository eing检eicht wird.	☐
d)	Die Tests werden nur in der Vorproduktionsumgebung ausgeführt, um die Produktionsumgebung zu simulieren.	☐

TAE-5.1.1 (K3) Der Lernende kann Testautomatisierung auf verschiedenen Teststufen innerhalb von Pipelines anwenden.

Begründung:

- a) FALSCH – da bei diesem Ansatz die Tests nicht als Quality Gates fungieren. Siehe Syllabus Kapitel 5.1.1 Absatz mit den Hauptansätzen, 2ter Hauptansatz 3ter Satz: “In diesem Fall fungieren Tests nicht als Quality Gate.” Wenn die Tests in einer separaten Pipeline ausgeführt werden, die nach einem erfolgreichen Deployment ausgelöst wird, können sie ein fehlgeschlagenes Deployment nicht verhindern. Laut Lehrplan, Kapitel 5.1.1, Aufzählung 4, Absatz, sind in diesem Fall andere, in der Regel manuelle Maßnahmen erforderlich, um ein fehlgeschlagenes Deployment rückgängig zu machen.
- b) KORREKT – Laut Lehrplan, Kapitel 5.1.1, siehe Aufzählung 4. Absatz können Systemintegrationstests als Teil des Deployments nach der Bereitstellung der Komponente ausgeführt werden. Dies kann von Vorteil sein, da aufgrund der Testergebnisse das Deployment fehlgeschlagen sein kann und auch wieder rückgängig gemacht werden kann. Dieser Ansatz ermöglicht es den Tests, als Quality Gates zu fungieren.
- c) FALSCH – da Systemintegrationstests typischerweise nicht in der lokalen Entwicklungsumgebung ausgeführt werden (siehe Lehrplan, Kapitel 2.1.2, Abschnitt “Integrationsumgebung”). Laut Lehrplan, Kapitel 5.1.1, Aufzählung im 2. Absatz werden in der lokalen Entwicklungsumgebung hauptsächlich Komponententests, GUI-Tests und API-Tests durchgeführt, nicht jedoch Systemintegrationstests.
- d) FALSCH – da Systemintegrationstests (laut Lehrplan, Kapitel 5.1.1, Aufzählung im 2. Absatz) oft Teil einer kontinuierlichen Delivery-Pipeline als Quality Gates sind und nicht ausschließlich in der Vorproduktionsumgebung ausgeführt werden. Die Vorproduktionsumgebung wird hauptsächlich zur Bewertung nicht-funktionaler Qualitätsmerkmale verwendet, siehe auch Lehrplan, Kapitel 2.1.2, Absatz “Vorproduktionsumgebung”.

Frage 21	TAE-5.1.2	K2	Punkte 1.0
----------	-----------	----	------------

Welche Aussage zum Konfigurationsmanagement in der Testautomatisierung ist AM EHESTEN zutreffend?

Wählen Sie EINE Option! (1 aus 4)

a)	Das Konfigurationsmanagement ermöglicht die Verwaltung von Testsuiten und Testumgebungskonfigurationen.	☒
b)	Das Konfigurationsmanagement verwaltet Testdaten und Testmittel immer unabhängig von der Software unter Test (SUT).	☐
c)	Das Konfigurationsmanagement ermöglicht die Verwaltung von Benutzerrechten für den Zugriff auf die Testautomatisierung.	☐
d)	Das Konfigurationsmanagement stellt die Ergebnisse der Testautomatisierung dar und ermöglicht somit eine leichte Analyse von Trends.	☐

TAE-5.1.2 (K2) Der Lernende kann Konfigurationsmanagement für Testmittel erklären.

Begründung:

- a) **KORREKT** – Testmittel und Testumgebungskonfigurationen können unter Konfigurationsmanagement gestellt und versioniert verwaltet werden, siehe Lehrplan, Kapitel 5.1.2, Abschnitt “Konfiguration der Testumgebung”.
- b) **FALSCH** – Testmittel werden durchaus auch abhängig von der SUT im Konfigurationsmanagement verwaltet, siehe Lehrplan, Kapitel 5.1.2, Abschnitt “Testsuiten/Testfälle”.
- c) **FALSCH** – Die Verwaltung von Benutzerrechten fällt in den Bereich des Benutzer- oder Rollenmanagements, nicht des Konfigurationsmanagements, siehe Lehrplan, Kapitel 5.1.2.
- d) **FALSCH** – Das Konfigurationsmanagement dient nicht der Analyse von Testergebnissen, sondern der Nachvollziehbarkeit und Reproduzierbarkeit von Testumgebungen und -daten, siehe Kapitel 5.1.2.

Frage 22	TAE-5.1.2	K2	Punkte 1.0
----------	-----------	----	------------

Welches der folgenden Elemente ist NICHT Teil der Testumgebungskonfiguration?

Wählen Sie EINE Option! (1 aus 4)

a)	Uniform Resource Locators (URLs)	☐
b)	Anmeldedaten	☐
c)	Testdaten	☐
d)	Gemeinsame Kernbibliothek	☒

TAE-5.1.2 (K2) Der Lernende kann Konfigurationsmanagement für Testmittel erklären.

Begründung:

- a) FALSCH – URLs gehören zur Konfiguration der Testumgebung, z. B. zur Definition von Zielsystemen, siehe Lehrplan, Kapitel 5.1.2, Abschnitt “Konfiguration der Testumgebung”.
- b) FALSCH – Anmeldedaten sind Teil der Testumgebung, da sie für die Authentifizierung im Testsystem benötigt werden, siehe Lehrplan, Kapitel 5.1.2, Abschnitt “Konfiguration der Testumgebung”.
- c) FALSCH – Testdaten sind ebenfalls ein Bestandteil der Testumgebungskonfiguration, siehe Lehrplan, Kapitel 5.1.2, Abschnitt “Testdaten”.
- d) **KORREKT** – Die gemeinsame Kernbibliothek gehört zum Testautomatisierungsframework TAF (Syllabus 3.1.3), In manchen Setups kann die Testumgebungskonfiguration in einer gemeinsamen Kernbibliothek abgelegt werden (Ablageort), sie ist aber weiterhin Konfiguration. Kapitel 5.1.2, Abschnitt “Konfiguration der Testumgebung”.

Frage 23	TAE-5.1.3	K2	Punkte 1.0
----------	-----------	----	------------

Was ist eine Grundlage für die Testautomatisierung von APIs?

Wählen Sie EINE Option! (1 aus 4)

a)	Die Nutzung von Shift-Left bei API-Tests, um Tests ausschließlich auf der höchsten Teststufe durchzuführen.	☐
b)	Die Best Practice, dass Komponentenintegrationstests die einzigen Tests sind, die für eine API-Infrastruktur durchgeführt werden sollten.	☐
c)	Die Annahme, dass API-Verbindungen keine Geschäftslogik-Beziehungen berücksichtigen müssen.	☐
d)	Die API-Dokumentation, die relevante Informationen wie Parameter, Kopfzeilen und Arten von Request-Response enthält.	☒

TAE-5.1.3 (K2) Der Lernende kann Abhängigkeiten bei der Testautomatisierung für eine API-Infrastruktur erläutern.

Begründung:

- a) FALSCH – Beim Shift-Left-Ansatz geht es darum so früh wie möglich zu testen, siehe ISTQB Glossar. Laut wird das Testen mit Shift-Left empfohlen, um Tests auf mehrere Teststufen aufzuteilen (siehe Lehrplan, Kapitel 5.1.3, zweiter Absatz).
- b) FALSCH – Der Lehrplan spricht davon API-Tests auf verschiedene Teststufen aufzuteilen und das eine Erweiterung der Tests durch Vertragstests, eine bewährte Praxis ist (siehe Lehrplan, Kapitel 5.1.3, zweiter Absatz).
- c) FALSCH – Im Lehrplan steht das Gegenteil, dass die Geschäftslogik und die Beziehungen zwischen den APIs sehr wohl relevant ist (siehe Lehrplan, Kapitel 5.1.3, Aufzählung).
- d) KORREKT – Die API-Dokumentation dient als Grundlage für die Testautomatisierung (siehe Lehrplan, Kapitel 5.1.3, Aufzählung).

Frage 24	TAE-5.1.3	K2	Punkte 1.0
----------	-----------	----	------------

Wie kann der Vertragstest dabei helfen Abhängigkeiten in der API-Intrastruktur aufzuzeigen?

Wählen Sie die beiden Aussagen zu Vertragstests, die zutreffend sind.

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Vertragstests ersetzen eine vollständige Schemavalidierung der Daten.	☐
b)	Vertragstests helfen bei einer frühen Identifikation von Fehlerzuständen im SDLC.	☒
c)	Vertragstests garantieren, dass alle API-Endpunkte jederzeit verfügbar sind.	☐
d)	Vertragstests helfen bei der Validierung von Interaktionen zwischen Diensten.	☒
e)	Vertragstests zeigen, dass der Anbieter immer die gleichen Ergebnisse liefert.	☐

TAE-5.1.3 (K2) Der Lernende kann Abhängigkeiten bei der Testautomatisierung für eine API-Infrastruktur erläutern.

Begründung:

- a) FALSCH – Der Vertragstest geht über die Schemavalidierung hinaus, ersetzt sie aber nicht und verlangt von beiden Kommunikationsparteien einen Konsens über die zulässigen Interaktionen, siehe Lehrplan, Kapitel 5.1.3, Abschnitt “Vertragstest”. Insofern stimmt es nicht, dass der Vertragstests eine Schemavalidierung ersetzt, da sie ein Bestandteil ist.
- b) KORREKT – Siehe Lehrplan, Kapitel 5.1.3, Abschnitt “Vertragstest”. Vertragstests helfen, Fehler früh im SDLC zu finden.
- c) FALSCH – Der Vertragstest erfasst die Interaktionen, die zwischen den einzelnen Diensten ausgetauscht werden, garantiert aber keine Verfügbarkeit von Endpunkten.“ siehe Lehrplan, Kapitel 5.1.3, Abschnitt “Vertragstest”.
- d) KORREKT – Laut Lehrplan, Kapitel 5.1.3, Abschnitt “Vertragstest”, wird die Validierung von Interaktionen unterstützt.
- e) FALSCH – Die Kommunikation steht im Fokus der Vertragstest und ob sich die Teilnehmer an vereinbarten Interaktionen halten, siehe Lehrplan, Kapitel 5.1.3, Abschnitt “Vertragstest”.

Frage 25	TAE-6.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Sie starten die Ausführung einer automatisierten Regressionstestsuite normalerweise am Ende des Arbeitstages, weil es ein länger laufender Test ist. Gelegentlich wird der Test bis zum Beginn des nächsten Werktags jedoch nicht abgeschlossen, obwohl er eigentlich innerhalb von fünf Stunden nach Beginn beendet sein sollte. Es scheint, dass einer oder mehrere Regressionstests deutlich länger benötigen als angenommen.

Wie lässt sich AM EFFIZIENTESTEN ermitteln, welche der Regressionstests diesen Zeitverzug verschulden?

Wählen Sie EINE Option! (1 aus 4)

a)	Sie starten den Regressionstest zu Beginn Ihres Arbeitstages und überwachen ihn manuell.	☐
b)	Sie automatisieren die Verifikationsfunktionen bis auf die tiefste technische Detailebene, sodass diese die tatsächlichen und erwarteten Ergebnisse automatisiert vergleichen und die jeweilige Abweichung hervorheben.	☐
c)	Sie implementieren eine zusätzliche Funktion in die Testprotokollierung, welche für jeden Testschritt, egal ob erfolgreich oder nicht, ein Screenshot automatisiert erzeugt.	☐
d)	Sie automatisieren die Erfassung des Startzeitpunktes, sowie der Dauer und ggf. des Endzeitpunktes der Tests innerhalb der Testautomatisierungslösung.	☒

TAE-6.1.1 (K3) Der Lernende kann Methoden zur Erfassung von Daten aus der Testautomatisierungslösung und dem System unter Test anwenden.

Begründung:

- a) FALSCH – Das wäre kein effizientes Vorgehen und würde Sie bei Ihrer täglichen Arbeit blockieren, deutlich effizienter wäre dagegen Antwort d).
- b) FALSCH – Die Implementierung von Vergleichsfunktionen ist zwar ein guter Ansatz und sollte, insbesondere bei komplexen Vergleichen, wenn möglich eingesetzt werden. (vgl. TAE-Lehrplan v2.0, Kapitel 6.1.1, Abschnitt „Features der Testautomatisierung, die Messungen und die Testberichterstellung unterstützen“, dritter Absatz), allerdings lässt sich so nicht herausfinden, welcher Test zu langsam läuft bzw. den Zeitverzug verschuldet. Eben weil der Fokus auf den Testfällen liegt, sollte auch hier in der Analyse angesetzt werden.
- c) FALSCH – Auch die Erfassung von Screenshots bei allen Testschritten ist ein probates Mittel sein, um Erkenntnisse über die automatisierte Testausführung zu dokumentieren, allerdings hilft es nicht, um den Zeitverzug zu erklären.
- d) **KORREKT – Die Erfassung der Start- und Endzeitpunkte, sowie der Dauer eines Tests ist in dieser Situation eine äußerst effiziente Möglichkeit rasch zu ermitteln, welcher der Tests zu langsam läuft bzw. den Zeitverzug verschuldet. Vgl. TAE-Lehrplan v2.0 Kapitel 6.1.1, Abschnitt TAS-Protokollierung, 1. Aufzählungspunkt.**

Frage 26	TAE-6.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Sie arbeiten an einem Projekt, in dem Sie verantwortlich sind, das bestehende Testautomatisierungsframework (TAF) für Webdiensttests, um erweiterte Testprotokollierungsfunktionen zu ergänzen. Ziel ist es, neben Dateiprotokollen auch einen HTML-Bericht zur schnellen Visualisierung der Testergebnisse zu erstellen.

Das SUT – ein Webdienst ohne Benutzeroberfläche – kommuniziert mit mehreren Legacy-Testsystemen. Leider sind die Tests sehr instabil. Sie sollen aussagekräftige Informationen in die Protokollierung aufnehmen, um die Fehlerursachen besser analysieren zu können.

Welche der folgenden Informationen sollten Sie AM EHESTEN in die Testprotokollierung aufnehmen – und warum?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Zeitstempel für die Testschritte, um zu erkennen, ob der Fehler mit einem Ausfall eines bestimmten Legacy-Systems zusammenhängt.	☒
b)	Screenshots, um die tatsächlichen Anfrage-Antwort-Paare zu dokumentieren.	☐
c)	Zufällig generierte Werte, um die Analyse der tatsächlichen Ergebnisse zu ermöglichen.	☒
d)	Ausgaben von Zählern, um leicht festzustellen, wie oft ein bestimmter Testfall ausgeführt wurde.	☐
e)	Skriptsprachen der verwendeten Testwerkzeuge, um Fehlerursachen in bestimmten Tests auszuschließen.	☐

TAE-6.1.1 (K3) Der Lernende kann Methoden zur Erfassung von Daten aus der Testautomatisierungslösung und dem System unter Test anwenden.

Begründung:

- a) **KORREKT** – Zeitstempel sind hilfreich, um Fehler zeitlich einzuordnen und mögliche Zusammenhänge mit Ausfällen von Legacy-Systemen zu erkennen, siehe Lehrplan, Kapitel 6.1.1, Aufzählung in Abschnitt “TAS-Protokollierung”.
- b) **FALSCH** – Da es sich um einen Webdienst ohne Benutzeroberfläche handelt, sind Screenshots nicht sinnvoll, siehe Aufgabentext. Stattdessen sollten Anforderungs- und Antwortdaten textuell protokolliert werden.
- c) **KORREKT** – Zufällig generierte Werte sollten unbedingt protokolliert werden, da sie zur späteren Reproduzierbarkeit und Fehlersuche notwendig sind, siehe Lehrplan, Kapitel 6.1.1, Aufzählung in Abschnitt “TAS-Protokollierung”.
- d) **FALSCH** – Sie wissen bereits das die Tests instabil sind, Zählerstände helfen Ihnen nicht dabei der Ursache dieser Fehler auf den Grund zu gehen. Die Zähler helfen vor allem bei Zuverlässigkeits- und Stresstests. Siehe Syllabus, Kapitel 6.1.1, Aufzählung in Abschnitt TAS-Protokollierung.
- e) **FALSCH** – Die Skriptsprachen sind in den Anforderungen an die TAS definiert. Zudem sollten Fehlerursachen durch Testwerkzeuge vor dem Einsatz der TAS verifiziert werden, siehe Lehrplan, Kapitel 7.1.1, “Die Verifizierung der Testautomatisierungsumgebung planen, einschließlich der Einrichtung der Testwerkzeuge”, erster Absatz. Daher haben die verwendeten Skriptsprachen für die Testprotokollierung keine Relevanz.

Frage 27	TAE-6.1.2	K4	Punkte 3.0
----------	-----------	----	------------

Sie sollen eine Testautomatisierungslösung für Last- und Stresstests eines zeitkritischen SUT entwerfen. Das System besteht aus mehreren Microservices, die jeweils strikten Anforderungen an ihr individuelles Antwortzeitverhalten genügen müssen. Ziel ist es, die Performanz sowohl transaktionsübergreifend (End-to-End) als auch pro Microservice zu messen und auswertbar zu machen.

Wie automatisieren Sie die Erhebung des Antwortzeitverhaltens so, dass die Ergebnisse möglichst einfach analysiert und korreliert werden können?

Wählen Sie EINE Option! (1 aus 4)

a)	Den Start- und Endzeitpunkt jeder Transaktion im Testprotokoll hinterlegen.	☐
b)	CPU- und Speicherauslastung der Microservices protokollieren, da diese Metriken zur Identifikation von Performanzproblemen ausreichen.	☐
c)	Für jede Transaktion eine Trace-ID vergeben und diese über alle Microservices hinweg durchreichen; alle gemessenen Zeitwerte mit der Trace-ID (und optional Span-IDs) verknüpfen, um Performanzdaten zu korrelieren.	☒
d)	Die Testprotokolle bewusst einfach halten und die Auswertung anschließend manuell durchführen.	☐

TAE-6.1.2 (K4) Der Lernende kann Daten aus der Testautomatisierungslösung (TAS) und dem System unter Test (SUT) analysieren, um die Ergebnisse besser zu verstehen.

Begründung:

- a) FALSCH – Eine rein aggregierte Betrachtung zeigt zwar Gesamtperformanz, aber keine Ursachen auf Microservice-Ebene. Anforderungen verlangen explizit beide Sichten (System + Microservice).
- b) FALSCH – Ressourcenauslastung kann Hinweis auf Engpässe geben, ist aber kein Ersatz für Messung von Antwortzeiten. CPU-Auslastung? Benutzer-perzipierte Performanz.
- c) **KORREKT – Durch den Einsatz von Trace-IDs kann die Performance über die beteiligten Microservices hinweg eindeutig für jede Transaktion zugeordnet und analysiert werden.**
- d) FALSCH – Manuelles Durchsuchen widerspricht dem Sinn einer automatisierten Protokollierung und skaliert nicht. Automatisierte Korrelation über Trace-IDs sind deutlich effizienter.

Frage 28	TAE-6.1.3	K2	Punkte 1.0
----------	-----------	----	------------

Welche wichtigen Informationen sollte ein Testfortschrittsbericht bei seiner Veröffentlichung typischerweise enthalten?

Wählen Sie EINE Option! (1 aus 4)

a)	Eine präzise Beschreibung der im Detail durchgeführten Testschritte je Testfall.	☐
b)	Informationen zur Entwicklungsumgebung, mit der das System unter Test entwickelt wurde.	☐
c)	Eine begründete Darstellung, warum bestimmte Tests fehlgeschlagen sind (Ursachen von Abweichungen).	☒
d)	Den Namen der Person, die den festgestellten Fehlerzustand korrigiert hat.	☐

TAE-6.1.3 (K2) Der Lernende kann erklären, wie ein Testfortschrittsbericht erstellt und veröffentlicht wird.

Begründung:

- a) FALSCH – Detaillierte Testschritte gehören in Testprotokolle/Testausführungsprotokolle, nicht in den Testfortschrittsbericht.
- b) FALSCH – Die Informationen zur Entwicklungsumgebung des Systems unter Test ist für den Testfortschrittsbericht irrelevant. Die Testumgebung ist ein wichtiger Punkt, der in einem Testausführungsbericht erwähnt werden sollte. Vgl. TAE-Lehrplan v2.0, Kapitel 6.1.3, Abschnitt Inhalt eines Testfortschrittsberichts, 1. Absatz.
- c) **KORREKT** – Ein Testfortschrittsbericht enthält eine Zusammenfassung der Ergebnisse einschließlich nicht bestandener Tests und deren Gründe/Abweichungen (Diagnose/Ursachen). Vgl. TAE-Lehrplan v2.0, Kapitel 6.1.3, Abschnitt Inhalt eines Testfortschrittsberichts, 2. Absatz.
- d) FALSCH – Die Person, die den Fehlerzustand korrigiert hat, ist keine sinnvolle Information für einen Testfortschrittsbericht. Die Korrektur eines Fehlerzustands erfolgt gemeinhin erst nach der Bereitstellung eines Testfortschrittsberichts auf Basis eines gemeldeten Fehlerberichts.

Frage 29	TAE-7.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Sie wurden zu einem Testautomatisierungsprojekt im Bankenbereich hinzugezogen, bei dem automatisierte Tests sporadisch fehlschlagen. Die wiederkehrende manuelle Prüfung der Testprotokolle kostet Zeit und führt dazu, dass das Vertrauen in die Testautomatisierung auf Leitungsebene gemindert wird. Nach genauerer Analyse fiel Ihnen auf, dass die nicht zuverlässig laufenden Tests mit einem externen Service interagieren, der die Validität von Transaktionen überprüft. Der Service ist nicht Teil des SUT, d. h., sein Verhalten wird nicht durch die Testfälle verifiziert. Sie stellen fest, dass aus Ihnen unbekannten Gründen, der externe Service zeitweise nicht zur Verfügung steht, was auch der Grund für die fehlgeschlagenen Tests ist.

Welcher der nachfolgenden Schritte ist für die Verifizierung der TAS in dieser Situation AM EHESTEN anzuwenden?

Wählen Sie EINE Option! (1 aus 4)

a)	Prüfen des Intrusionsgrads der Testskripte, um auszuschließen, dass zu niedrige Intrusion für die technischen Fehler verantwortlich ist.	☐
b)	Anmeldung beim externen Service als Teil der Vorbedingungen eines Testfalls, um sicherzugehen, dass der Service für die Testausführung verfügbar ist.	☒
c)	Überprüfen, dass die TAS aus einem zentralen Installationsrepositorium in der Testumgebung installiert wurde, um sicherzugehen, dass die letzte Version bzw. Konfiguration der TAS verwendet wird.	☐
d)	Konfigurationsmanagement einsetzen, um die Konsistenz von Testdaten und Testskripte über verschiedene Versionen hinweg sicherzustellen.	☐

TAE-7.1.1 (K3) Der Lernende kann die Verifizierung der Testautomatisierungsumgebung planen, einschließlich der Einrichtung der Testwerkzeuge.

Begründung:

- a) FALSCH – Es ist in dem gegebenen Szenario nicht ersichtlich, dass ein zu hoher Intrusionsgrad ein Grund für die technischen Fehler war. Zudem ist die Begründung grundlegend falsch. Ein niedriger Grad an Intrusion ist üblicherweise nicht für Fehlerwirkungen oder Fehlschläge bzw. technische Fehler verantwortlich. Vgl. TAE-Lehrplan v2.0, Kapitel 7.1.2, Abschnitt Berücksichtigung der Intrusivität von Testautomatisierungswerkzeugen: „Die TAS ist oft eng mit dem SUT gekoppelt. [...] Eine solche enge Integration kann jedoch auch zu nachteiligen Ergebnissen führen. [...] Ein hoher Grad an Intrusion kann Fehlerwirkungen beim Testen hervorrufen, die in der Produktionsumgebung nicht auftreten.“
- b) KORREKT – Bevor Testfälle mit externen Services bzw. Servern ausgeführt werden, ist es eine bewährte Vorgehensweise zu überprüfen, ob die externen Services bzw. Server auch verfügbar sind (vgl. TAE-Lehrplan v2.0, Kapitel 7.1.1, Abschnitt Konnektivität der internen und externen Systeme/Schnittstellen: „Es ist zum Beispiel eine bewährte Vorgehensweise, sich bei Servern anzumelden [...]. Die Festlegung von Vorbedingungen für die Testautomatisierung ist wichtig, um sicherzustellen, dass die TAS korrekt installiert und konfiguriert wurde. In diesem Fall würden die Testfälle bei Nicht-Verfügbarkeit zwar ebenfalls fehlschlagen, allerdings bereits schon in den Vorbedingungen. Gegebenenfalls wäre es sogar möglich, auf die Nichtverfügbarkeit zu reagieren – etwa durch die Nutzung eines alternativen oder virtualisierten Dienstes. In jedem Fall verfügt Testteam damit über ein Mittel, um Tests, die aufgrund der Nichtverfügbarkeit fehlschlagen würden, frühzeitig zu erkennen.“
- c) FALSCH – Die Installation der TAS in die Testumgebung ist grundsätzlich eine sehr bewährte und empfohlene Vorgehensweise (vgl. TAE-Lehrplan v2.0, Kapitel 7.1.1, Abschnitt Installation, Einrichtung, Konfiguration und Anpassung von Testwerkzeugen, 2. Absatz: „Eine automatisierte Installation oder auch das Kopieren aus einem zentralen Repository hat Vorteile. Dies garantiert, dass Tests auf unterschiedlichen SUTs mit der gleichen Version und der gleichen Konfiguration der TAS durchgeführt werden, falls dies gefordert wird.“) allerdings steht die korrekte Installation nicht im Zusammenhang mit sporadisch auftretenden Nichtverfügbarkeit des externen Service.
- d) FALSCH – Konfigurationsmanagement ist zwar hilfreich und eine bewährte Vorgehensweise, allerdings hat Konfigurationsmanagement nichts mit der Aufgabenstellung und der sporadischen Nicht-Verfügbarkeit des externen Service zu tun.

Frage 30	TAE-7.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Sie arbeiten in einem Softwareentwicklungsteam, das Tests in unterschiedlichen Testumgebungen durchführen muss. Ihr Vorgesetzter hat festgestellt, dass viel Zeit darauf verwendet wird, falsch-positive Ergebnisse zu analysieren, weil die Testautomatisierungslösung (TAS) in neuen Umgebungen nicht korrekt konfiguriert ist. Zudem gibt es Versionsunterschiede zwischen den TAS-Installationen in den einzelnen Umgebungen.

Welche beiden der folgenden Maßnahmen wären **AM BESTEN** geeignet, diese Situation zu verbessern?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Sie erstellen ein automatisiertes Installationsskript für die Werkzeuge und Konfigurationen, aus denen die TAS besteht.	☒
b)	Sie beschränken die Nutzung der TAS auf die Testumgebungen, auf denen die Konfiguration korrekt erstellt wurde und funktioniert.	☐
c)	Sie nutzen ein zentrales Repository zum Speichern der TAS, auf das alle Testumgebungen zugreifen können.	☒
d)	Sie führen manuelle Tests durch, um zu überprüfen, ob die TAS in allen Testumgebungen korrekt eingerichtet ist.	☐
e)	Sie überspringen aus Zeitgründen die Implementierung von Komponententests für die TAS.	☐

TAE-7.1.1 (K3) Der Lernende kann die Verifizierung der Testautomatisierungsumgebung planen, einschließlich der Einrichtung der Testwerkzeuge.

Begründung:

- a) **KORREKT** – Automatisierte Installations- und Konfigurationsskripte stellen sicher, dass die TAS in allen Umgebungen konsistent eingerichtet wird (siehe Lehrplan, Kapitel 7.1.1, Absatz “Installation, Einrichtung, Konfiguration und Anpassung von Testwerkzeugen”).
- b) **FALSCH** – Die TAS sollte gerade so gestaltet sein, dass sie in verschiedenen Testumgebungen zuverlässig funktioniert (siehe Lehrplan, u. a. Kapitel 7.1.1, Absatz “Installation, Einrichtung, Konfiguration und Anpassung von Testwerkzeugen”). Einfach Testumgebungen auszusparen ist keine gute Option.
- c) **KORREKT** – Ein zentrales Repository ermöglicht die Verwendung einheitlicher TAS-Versionen und verhindert veraltete Installationen (siehe Lehrplan, Kapitel 7.1.1, Absatz “Installation, Einrichtung, Konfiguration und Anpassung von Testwerkzeugen”).
- d) **FALSCH** – Die Konfigurationen sind nicht korrekt und es gibt Versionsunterschiede bei der TAS auf den unterschiedlichen Umgebungen, ist laut Aufgabenstellung bereits bekannt. Manuelle Prüfungen sind zudem nicht skalierbar und beheben auch nicht das zugrundeliegende Problem der inkonsistenten Einrichtung.
- e) **FALSCH** – Komponententests sind wichtig und üblich, um Probleme frühzeitig zu erkennen – ihr Verzicht kann zu instabiler Funktionalität führen (siehe Lehrplan, Kapitel 7.1.1, Absatz “TAF-Komponententests”).

Frage 31	TAE-7.1.2	K2	Punkte 1.0
----------	-----------	----	------------

Sie sind dabei, eine automatisierte Testsuite zu verifizieren. Während dieses Prozesses stellen Sie fest, dass einige Testskripte bei einem Durchlauf auf ihrer Testumgebung erfolgreich sind, beim nächsten Durchlauf jedoch fehlschlagen – sie liefern also keine zuverlässigen Ergebnisse.

Was sollten Sie als ERSTES tun, um die Gültigkeit dieser Testskripte zu überprüfen?

Wählen Sie EINE Option! (1 aus 4)

a)	Sie führen die Tests manuell aus, um Fehlerwirkungen durch die Intrusivität der Testwerkzeuge auszuschließen.	☐
b)	Sie führen die automatisierte Testsuite erneut auf einem frisch gestarteten System aus und analysieren die Ergebnisse.	☐
c)	Sie isolieren die betroffenen Testskripte und analysieren Sie sie separat.	☒
d)	Sie prüfen die betroffenen Testskripte mittels statischer Code-Analysen.	☐

TAE-7.1.2 (K2) Der Lernende kann das korrekte Verhalten für ein gegebenes automatisiertes Testskript und/oder eine Testsuite erläutern.

Begründung:

- a) FALSCH – Die Intrusivität macht sich vor allem durch unterschiedliche Ausführungsergebnisse zwischen Test- und Produktionsumgebung bemerkbar, siehe Lehrplan, Kapitel 7.1.2, “Berücksichtigung der Intrusivität von Testautomatisierungswerkzeugen”. Laut Aufgabenstellung liefen die Tests aber auf der gleichen Plattform mit unterschiedlichen Ergebnissen.
- b) FALSCH – Ein reiner Wiederholungslauf liefert keine neuen Erkenntnisse, wenn die Ursache nicht untersucht wird und erzeugt nur zusätzlichen Aufwand. Die unzuverlässigen Tests sollten besser separat analysiert werden, siehe Lehrplan, Kapitel 7.1.2, Absatz “Berücksichtigung der Wiederholbarkeit von Tests”.
- c) KORREKT – Das gezielte Isolieren und Analysieren der instabilen Testskripte ist der erste sinnvolle Schritt zur Fehleridentifikation, siehe Lehrplan, Kapitel 7.1.2, Absatz “Berücksichtigung der Wiederholbarkeit von Tests”.
- d) FALSCH – Statische Analysen können proaktiv zur Qualitätssicherung eingesetzt werden (siehe Lehrplan, Kapitel 7.1.4). Als ersten Schritt wäre es ratsamer die betroffenen Tests zu isolieren und separat zu analysieren, siehe Lehrplan, Kapitel 7.1.2, Absatz “Berücksichtigung der Wiederholbarkeit von Tests”.

Frage 32	TAE-7.1.2	K2	Punkte 1.0
----------	-----------	----	------------

Sie erweitern eine API-basierte Testautomatisierungslösung (TAS) für eine Webapplikation auf die grafische Benutzungsschnittstelle (GUI). Dazu haben Sie ein kommerzielles GUI-Testautomatisierungswerkzeug angeschafft und erfolgreich in die TAS eingebunden. Nach einigen Testläufen stellen Sie fest, dass einige Tests unzuverlässig laufen und Funktionen, die beim API-Test erfolgreich getestet werden beim GUI-Test fehlschlagen. Um falsch-positive Ergebnisse auszuschließen, führen Sie und Ihr Team die fehlgeschlagenen GUI-Tests manuell aus und stellen erstaunt fest, dass alle Tests bestanden wurden. Sie prüfen daraufhin die Konfiguration des neuen GUI-Testautomatisierungswerkzeug. Dort stellen Sie fest, dass der dynamische Wartemechanismus für die Steuerungen und Beobachtungen der Steuerelemente deaktiviert ist, d. h., die Kommandos werden von dem GUI-Testautomatisierungswerkzeug sofort an das SUT abgesetzt.

Welche der nachfolgenden Vermutungen ist AM EHESTEN zutreffend?

Wählen Sie EINE Option! (1 aus 4)

a)	Die Testskripte verfügen über zu wenig Verifikationspunkte.	☐
b)	Die enge Kopplung zwischen TAS und SUT über die GUI sorgt für Race Conditions im SUT.	☐
c)	Die Konfiguration der Testumgebung muss geprüft werden.	☐
d)	Die Intrusion des GUI-Testautomatisierungswerkzeugs ist zu hoch.	☒

TAE-7.1.2 (K2) Der Lernende kann das korrekte Verhalten für ein gegebenes automatisiertes Testskript und/oder eine Testsuite erläutern.

Begründung:

- a) FALSCH – Es gibt in diesem Szenario keinen Hinweis darauf, dass nicht ausreichend Verifikationspunkte vorhanden sind. Im Gegenteil: die vorhandenen Verifikationspunkte liefern Ergebnisse – nur eben unzuverlässig.
- b) FALSCH – Es gibt in dem Szenario keinen Hinweis darauf, dass es sich bei dem System um ein nebenläufiges bzw. verteiltes System handelt. Daher ist auch die Vermutung von race conditions nicht naheliegend.
- c) FALSCH – Die Testumgebung ist in beiden Fällen dieselbe. Es geht hier lediglich um einen GUI-basierten Test für eine Webapplikation. Daher kann das keine Ursache sein.
- d) KORREKT – Ein automatisierter GUI-Test bzw. ein GUI-Testwerkzeug greift in das Verhalten eines SUTs ein, indem es mit den UI-Steuerungselementen mithilfe von Softwarefunktionen interagiert. D.h., das SUT wird grundsätzlich anders bedient als in der späteren Produktivumgebung. Dadurch, dass die Kommandos an die UI-Steuerungselemente ohne Wartezeit an das SUT abgesetzt werden, kann es leicht dazu kommen, dass das SUT es schlichtweg zeitlich nicht schafft, diese Kommandos gleichschnell abzuarbeiten. Als Resultat kann es dazu kommen, dass UI-Steuerungselemente, Dialoge oder ganze Seite noch gar nicht erschienen sind, der Testfall aber bereits eine Eigenschaft von diesen Komponenten auslesen oder eine Aktion auf einem nicht vorhandenen UI-Steuerungselement ausführen möchte. Dies führt zu sporadisch auftretenden Fehlschlägen, nicht jedoch, wenn die Testfälle manuell ausgeführt werden. Vgl. TAE-Lehrplan v2.0, Kapitel 7.1.2, Abschnitt Berücksichtigung der Intrusivität von Testautomatisierungswerkzeugen.

Frage 33	TAE-7.1.3	K2	Punkte 1.0
----------	-----------	----	------------

Sie arbeiten an einem Projekt zur Automatisierung einer Regressionstestsuite für ein System unter Test (SUT). Bei der letzten manuellen Ausführung wurden alle Tests bestanden. Wenn Sie dieselbe Suite jedoch über die Testautomatisierungslösung (TAS) ausführen, schlagen einige Tests fehl.

Was sollten Sie in dieser Situation am EHESTEN tun?

Wählen Sie EINE Option! (1 aus 4)

a)	Analysieren Sie die Protokolldateien des SUT und der TAF, um die Ursache des Problems zu ermitteln.	☒
b)	Führen Sie eine Schemavalidierung durch, um die Datentypen und Wertebereiche ihrer Daten zu prüfen.	☐
c)	Erstellen Sie einen Fehlerbericht für das SUT, um das Problem zu dokumentieren.	☐
d)	Ignorieren Sie das Verhalten, da automatisierte Tests sich naturgemäß anders verhalten als manuelle Tests.	☐

TAE-7.1.3 (K2) Der Lernende kann identifizieren, wo die Testautomatisierung unerwartete Ergebnisse liefert.

Begründung:

- a) **KORREKT** – Die Analyse der Protokolle hilft, die Ursache der fehlgeschlagenen Tests zu identifizieren. siehe Lehrplan, Kapitel 7.1.3, erster Absatz, zweiter Satz: „Prüfung der Testprotokolle...“.
- b) **FALSCH** – Laut Aufgabenstellung scheint es bei manueller Ausführung der Test keine Probleme zu geben, daher wäre naheliegender erst einmal die Protokolldateien auf Probleme hin zu untersuchen.
- c) **FALSCH** – Ein Fehlerbericht enthält die Doku des Auftretens, der Art und Status und wird nur erstellt werden, wenn der Fehlerzustand im SUT analysiert wurde. Laut der Aufgabenbeschreibung laufen die Tests manuell, aber erfolgreich durch (siehe ISTQB Glossar zu Fehlerbericht).
- d) **FALSCH** – Ein solches Verhalten widerspricht dem Zweck der Testautomatisierung und muss analysiert werden. Siehe Syllabus, Kapitel 1.1.1 Vorteile der Testautomatisierung (verbesserte Konsistenz über Zyklen hinweg, weniger anfällig für menschliche Fehlhandlungen, ...)

Frage 34	TAE-7.1.4	K2	Punkte 1.0
----------	-----------	----	------------

Sie bereiten die Ausführung einer Testautomatisierungssuite für eine sicherheitskritische Anwendung vor, die höchsten Sicherheitsanforderungen genügen muss.

Welche Maßnahme sollten Sie am EHESTEN wählen, um den Code der Testautomatisierung gegenüber möglichen Sicherheitsmängeln robuster zu machen?

Wählen Sie EINE Option! (1 aus 4)

a)	Sie führen eine automatisierte statische Analyse der Testprotokolle nach möglichen Anmeldedaten aus.	☐
b)	Sie verwenden in ihren Testfällen gängige Entwurfsmuster, um Sicherheitsmängel auszuschließen.	☐
c)	Sie führen die Tests der Testsuite isoliert und methodisch aus, um potenzielle Sicherheitslücken zu erkennen.	☐
d)	Sie verwenden ein Werkzeug zur statischen Analyse, um Sicherheitslücken zu identifizieren.	☒

TAE-7.1.4 (K2) Der Lernende kann erläutern, wie statische Analyse helfen kann, die Qualität des Codes für die Testautomatisierung zu verbessern.

Begründung:

- a) FALSCH – Es sollte der Quellcode der Testautomatisierung statisch analysiert werden, siehe Lehrplan, Kapitel 7.1.4, erster Absatz.
- b) FALSCH – Die Verwendung von Entwurfsmustern ist eine Maßnahme die Wartbarkeit ihrer Testautomatisierung zu erhöhen (siehe Lehrplan, Kapitel 4.3.1), sie ist aber keine gute Maßnahme, um Sicherheitsmängel zu prüfen oder auszuschließen.
- c) FALSCH – Eine isolierte Ausführung führt nicht automatisch zur Aufdeckung von Sicherheitslücken, sondern ist eher eine Maßnahme, um Testfälle zu prüfen, die unerwartete Ergebnisse liefern, siehe Lehrplan, Kapitel 7.1.3).

d) KORREKT – Statische Analysewerkzeuge helfen, potenzielle Sicherheitsschwachstellen im Code frühzeitig zu identifizieren, (siehe Lehrplan, Kapitel, 7.1.4, letzter Absatz).

Frage 35	TAE-8.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Sie arbeiten an einem Testautomatisierungsprojekt zur Automatisierung der Tests der grafischen Benutzeroberfläche (GUI) eines Online-Webshops. Der Webshop enthält einen Assistenten, der Benutzern bei der Einrichtung ihrer Konten hilft – einschließlich Name, Rechnungs- und Lieferadresse sowie Sicherheitsdaten.

Derzeit befindet sich die Entwicklung der Software in einer Phase, in der Usability-Tester den Assistenten prüfen und iterativ Empfehlungen für Änderungen abgeben: Die Entwickler ändern die grafische Benutzeroberfläche, die Usability-Tester prüfen die Änderungen und wiederholen ihre Tests. Die Testautomatisierung konzentriert sich dabei auf Wartungstests. In diesen GUI-basierten Tests enthalten die Testdaten auch UI-Locator-Werte. Ein bestehendes Problem ist, dass Entwickler häufig die internen Bezeichner der UI-Elemente ändern, was den Wartungsaufwand für die Tests stark erhöht.

Welche der folgenden Maßnahmen bietet die BESTE Verbesserung?

Wählen Sie EINE Option! (1 aus 4)

a)	Führen Sie eine Schema-Validierung durch, um sicherzustellen, dass alle erforderlichen GUI-Elemente vorhanden sind.	☐
b)	Verbessern Sie die Testprotokollierung, um Informationen über UI-Elemente und ihre Locator-Werte einzuschließen.	☐
c)	Erstellen Sie ein Test-Histogramm, das es TAEs ermöglicht, besonders fehleranfällige Testfälle zu identifizieren.	☐
d)	Verwenden Sie einen KI-basierten Algorithmus mit Bilderkennung, um veränderte Selektoren zu erkennen.	☒

TAE-8.1.1 (K3) Der Lernende kann Möglichkeiten zur Verbesserung von Testfällen durch Datensammlung und Datenanalyse entdecken.

Begründung:

- a) FALSCH – Schema-Validierung ist eher für API-Tests geeignet und hilft nicht bei der Stabilisierung von GUI-Locator-Werten (siehe Lehrplan, Kapitel 8.1.1, Abschnitt “Schemavalidierung”)
- b) FALSCH – Verbesserte Protokollierung erleichtert die Fehleranalyse, löst aber nicht das Locator-Problem, das geht am ehesten mit KI, siehe Lehrplan, Kapitel 8.1.1, Abschnitt “Künstliche Intelligenz”.
- c) FALSCH – Ein Histogramm zeigt anfällige Testfälle, reduziert aber nicht den Wartungsaufwand, es ist eher geeignet, um unzuverlässige Tests aufzudecken, siehe Lehrplan 8.1.1, Abschnitt “Testhistogramm”.
- d) **KORREKT – KI-gestützte Lokator-Erkennung und Selbstheilung helfen, den Aufwand bei häufigen UI-Änderungen zu verringern siehe Lehrplan, Kapitel 8.1.1, Abschnitt “Künstliche Intelligenz”.**

Frage 36	TAE-8.1.1	K3	Punkte 2.0
----------	-----------	----	------------

Ihr Unternehmen verfügt über eine Regressionstestsuite mit mehr als 1000 automatisierten Testfällen, die sich über die Jahre als äußerst zuverlässig erwiesen hat. Kürzlich hat das Entwicklungsteam beschlossen, den Technologie-Stack zu modernisieren und überarbeitet derzeit die Funktionsweise des Frontends. Sie stellen fest, dass die Anwendung nun deutlich stärker API-gesteuert ist als zuvor, was Auswirkungen auf die Darstellung der UI-Elemente hat. Sie gehen davon aus, dass etwa 75 % Ihrer automatisierten Testfälle davon betroffen sein könnten.

Wie gehen Sie **AM BESTEN** bei der Analyse und Beseitigung der Auswirkungen durch die beschriebenen Umstellungen vor?

Wählen Sie **EINE** Option! (1 aus 4)

a)	Führen Sie die Testfälle mehrfach in einer CI/CD-Pipeline aus, analysieren Sie die Ergebnisse visuell und leiten Sie Erkenntnisse aus einem Test-Histogramm ab.	☐
b)	Nutzen Sie Schemavalidierungswerkzeuge, um Schemaverletzungen durch die neue API auszuschließen und zu beheben.	☒
c)	Führen Sie keine Datenanalyse durch und erstellen stattdessen die automatisierten Testfälle neu, die Aufgrund der Umstellung nicht ordnungsgemäß funktionieren.	☐
d)	Schreiben Sie Assertions, um zu prüfen, ob die Parameter den Vorgaben entsprechen und nicht NULL sind.	☐

TAE-8.1.1 (K3) Der Lernende kann Möglichkeiten zur Verbesserung von Testfällen durch Datensammlung und Datenanalyse entdecken.

Begründung:

- a) FALSCH – Die Analyse von Test-Histogrammen bei über 1000 Testfällen ist sehr zeitaufwendig und eher dazu geeignet, um unzuverlässige Tests aufzudecken, siehe Lehrplan 8.1.1, Abschnitt “Testhistogramm”. Zudem lässt sich der Einfluss auf die Testfälle bereits aus der technischen Umstellung ableiten, siehe Aufgabenstellung.
- b) **KORREKT – API-Schema-Validierungstools ermöglichen eine schnelle Prüfung von Änderungen auf API-Ebene, siehe Lehrplan, Kapitel 8.1.1, Abschnitt “Schemavalidierung”.**
- c) FALSCH – Das vollständige Neuerstellen von 75 % der Testfälle ist nicht effizient, solange bessere Alternativen (Lehrplan, Kapitel 8.1.1) existieren.
- d) FALSCH – Eine Schemavalidierung, macht es überflüssig einzelne Assertions zu schreiben, um Elemente zu prüfen, weil das die Werkzeuge übernehmen (siehe Lehrplan, Kapitel 8.1.1, Abschnitt “Schemavalidierung”).

Frage 37	TAE-8.1.2	K4	Punkte 3.0
----------	-----------	----	------------

Sie arbeiten an einer automatisierten Regressionstest-Suite, deren Ausführung aktuell zu lange dauert und die über Nacht nicht vollständig abgeschlossen werden kann. Die Testumgebung steht dabei ausschließlich nachts für Regressionstests zur Verfügung. Eine parallele Ausführung der Tests ist nicht möglich, da das Zielsystem teuer ist und nur in einer einzigen Instanz vorliegt.

Was sollten AM BESTEN Ihre nächsten Schritte sein, um sicherzustellen, dass die Ausführung der Tests über Nacht vollständig abgeschlossen wird, ohne den Abdeckungsgrad der Tests zu verringern?

Wählen Sie ZWEI Optionen! (2 aus 5)

a)	Teilen Sie die Regressionstest-Suite auf und führen Sie Teilmengen an verschiedenen Nächten aus.	☐
b)	Zentralisieren/entkoppeln Sie Setup- und Teardown-Methoden auf Suite-/Fixture-Ebene, um redundante Initialisierung zu vermeiden und die Laufzeit zu reduzieren.	☒
c)	Ersetzen Sie ereignis-/bedingungs-basierte Warte- bzw. Polling-Mechanismen durch hartcodierte feste Wartezeiten, um die Ausführung zu vereinfachen.	☐
d)	Entfernen Sie besonders lange Tests aus der Suite, um die Gesamtlaufzeit zu verringern.	☐
e)	Identifizieren und entfernen Sie überlappende bzw. duplizierte Tests, sofern sie keinen zusätzlichen Abdeckungsbeitrag liefern (nachweisbar z. B. über eine Abdeckungsmatrix).	☒

TAE-8.1.2 (K4) Der Lernende kann technische Aspekte einer eingesetzten Testautomatisierungslösung (TAS) analysieren und Verbesserungen empfehlen.

Begründung:

- a) FALSCH – Das Aufteilen der Testsuite auf mehrere Nächte ermöglicht NICHT die vollständige Testausführung innerhalb des verfügbaren Zeitfensters über Nacht, siehe Lehrplan, Kapitel 8.1.2, Abschnitt “Testausführung”.
- b) KORREKT – Suite/Fixture-basiertes Setup/Teardown-Methoden vermeidet Mehrfach-Initialisierung und reduziert Laufzeiten ohne Abdeckungsverlust (vgl. Optimierung, siehe Lehrplan, Kapitel 8.1.2, Absatz Setup und Teardown).
- c) FALSCH – Feste Wartezeiten sind unflexibel und typischerweise langsamer/unzuverlässiger als bedingte/ereignisbasiertes Setup/Teardown Waits (siehe Lehrplan, Kapitel 8.1.2 unter Aufzählungspunkt “Evaluierung von Wartemechanismen”)
- d) FALSCH – Das Entfernen von langsamen Tests reduziert die Abdeckung, was laut Aufgabenstellung keine Option bzw. unzulässig ist, und birgt das Risiko, dass Fehler in die Produktion gelangen.
- e) KORREKT – Das Entfernen von redundanten Tests kann die Gesamtlaufzeit der Testsuite reduzieren, siehe Lehrplan, Kapitel 8.1.2, Abschnitt “Testausführung”.

Frage 38	TAE-8.1.2	K4	Punkte 3.0
----------	-----------	----	------------

Sie wurden als TAE beratend zu einem Testautomatisierungsprojekt hinzugezogen, bei dem die Testautomatisierer sich über nicht mehr kontrollierbarer Inkonsistenzen bei der Wiederherstellung des SUT nach einer Fehlerwirkung beschwerten. Sie haben die Testfälle sorgfältig analysiert und festgestellt, dass eine Vielzahl von Methoden implementiert wurden, um die Wiederherstellung des SUT nach Fehlern zu handhaben. Teilweise ist die Fehlerbehandlung Teil des automatisierten Testfalls, teilweise Teil des Setups des Testfalls und der Testsuite, andere Testfälle verwenden separat bereitgestellte Schnittstellen, um direkt mit dem SUT zu interagieren. Diese Variabilität macht die Analyse und Anpassung der Testfälle zeitaufwändig und wirkt sich daher negativ auf den Wartungsaufwand generell aus.

Welche Verbesserung würden Sie in dieser Situation AM EHESTEN empfehlen?

Wählen Sie EINE Option! (1 aus 4)

a)	Etablieren eines einheitlichen Fehlerwiederherstellungsprozesses in der TAS und Einbindung in die zentralen Funktionen der Testautomatisierungslösung.	☒
b)	Wechsel zu einem schlüsselwortgetriebenen Ansatz mit der Wiederherstellung als eines der Schlüsselwörter.	☐
c)	Anlegen einer Bibliothek mit verschiedenen Funktionen für die Wiederherstellung des SUT, damit eine bessere skriptübergreifende Wiederverwendung gegeben ist.	☐
d)	Auslagern der aktuellen Mechanismen für die SUT-Wiederherstellung in die Setup- bzw. Teardown-Methoden der Testfälle.	☐

TAE-8.1.2 (K4) Der Lernende kann technische Aspekte einer eingesetzten Testautomatisierungslösung analysieren und Verbesserungen empfehlen.

Begründung:

- a) **KORREKT** – Die Implementierung eines einheitlichen Fehlerwiederherstellungsprozesses in den zentralen Funktionsbibliotheken der Testautomatisierungslösung bringt in dieser Situation den größten Nutzen, da diese Verbesserung den Inkonsistenzen entgegenwirkt und zu einer Vereinheitlichung der Testskripte beiträgt. Dadurch, dass der Fehlerwiederherstellungsprozess in die zentralen Funktionsbibliotheken der Testautomatisierungslösung eingebunden wird, steht der Fehlerwiederherstellungsmechanismus allen Testskripten gleichermaßen zur Verfügung und kann durchgehend eingesetzt werden. Vgl. TAE-Lehrplan, Kapitel 8.1.2, Abschnitt Skripterstellung, zweiter Absatz, zweiter Listenelement.
- b) **FALSCH** – Eine Umstellung des Automatisierungsansatzes für die einheitliche Verwendung eines Wiederherstellungsprozesses ist nicht erforderlich. Zum einen bewirkt die Bereitstellung von Schlüsselwörtern der aktuellen Mechanismen keine Vereinheitlichung, zum anderen sind diese auch nicht in die zentralen Funktionsbibliotheken eingebunden. Der Testautomatisierungsansatz kann zwar grundsätzlich auch geändert werden (vgl. Lehrplan, Kapitel 8.1.2, Abschnitt “Skripterstellung”, 1. Absatz), dies erzeugt aber einen sehr hohen Aufwand, Antwort a) bringt den gewünschten Effekt mit weniger Aufwand.
- c) **FALSCH** – Die Implementierung von verschiedenen Funktionen in einer Bibliothek ändert nichts an dem Problem, der inkonsistenten Verwendung der Fehlerwiederherstellung des SUT. Es erfolgt keine Konsolidierung bzw. Vereinheitlichung der Testskripte im Gegensatz zu Antwort a).
- d) **FALSCH** – Die Verwendung von Setup- und Teardown-Methoden ist generell eine empfohlene Vorgehensweise, um die Wartbarkeit von Testfällen zu vereinfachen, da wiederkehrende Aktionen die vor oder nach jedem Testskript / jeder Testsuite wiederholt werden, ausgelagert und übergreifend verwendet werden (vgl. TAE-Lehrplan, Kapitel 8.1.2 Abschnitt Setup und Teardown, durch die alleinige Auslagerung der aktuellen Mechanismen wird das Problem der Variabilität und Vielzahl der Wiederherstellungsmethoden an diversen Stellen nicht gelöst!

Frage 39	TAE-8.1.3	K3	Punkte 2.0
----------	-----------	----	------------

Sie sind als Test Automation Engineer (TAE) für die Testautomatisierung einer Webanwendung verantwortlich. Vor Kurzem wurde ein neues Release des Testobjekts (SUT) mit umfangreichen Änderungen an Front- und Backend veröffentlicht. Sie sollen beurteilen, ob die bestehende Testautomatisierungslösung (TAS) und die automatisierten Testmittel angepasst werden müssen.

Welche der nachfolgenden Aktivitätsabfolgen ist in dieser Situation am BESTEN geeignet?

Wählen Sie EINE Option! (1 aus 4)

a)	Durchführen einer Änderungsauswirkungsanalyse; Planung der Umsetzung der Anpassungen in der TAA; schrittweise Umsetzung der Anpassungen inklusive Smoke-Tests; Durchführung eines vollständigen Regressionstests; gegebenenfalls Ursachenanalyse festgestellter Fehlerwirkungen.	☒
b)	Durchführen einer Änderungsauswirkungsanalyse, schrittweise Umsetzung der Anpassungen in der TAA; Durchführung ausgewählter funktionaler Regressionstests; Zurücksetzen der Änderungen bei fehlgeschlagenen Testfällen; gegebenenfalls Ursachenanalyse.	☐
c)	Durchführen einer Änderungsauswirkungsanalyse, Planung der Umsetzung der Anpassungen in der TAA; Umsetzung aller Anpassungen inklusive eines vollständigen Regressionstest, Freigabe der TAS für den Einsatz an den Testmanager; gegebenenfalls Ursachenanalyse.	☐
d)	Durchführen einer Änderungsauswirkungsanalyse; Planung der Umsetzung der Anpassungen in der TAA; schrittweise Umsetzung der Anpassungen inklusive Smoke-Tests; Durchführung eines vollständigen Regressionstests; Zurücksetzen der Änderungen bei fehlgeschlagenen Testfällen. Ein fehlgeschlagener Test ist nicht zwingend auf eine Fehlerwirkung in der TAS zurückzuführen.	☐

TAE-8.1.3 (K3) Der Lernende kann automatisierte Testmittel zwecks Anpassung an Updates des Systems unter Test restrukturieren.

Begründung:

- a) **KORREKT** – Diese Abfolge an Aktivitäten umfasst die im Lehrplan empfohlene Vorgehensweise um Änderungen an den Komponenten der Testumgebung, der Testautomatisierungslösung, den automatisierten Testmitteln zu identifizieren, umzusetzen und qualitätszusichern. „Durchführen einer Änderungsauswirkungsanalyse“ --> Vgl. TAE-Lehrplan v2.0. Kapitel 8.1.2, Abschnitt Identifizieren von Änderungen an den Komponenten der Testumgebung: „Es ist zu beurteilen, welche Änderungen und Verbesserungen vorgenommen werden müssen und ob dadurch Änderungen an den Testmitteln, den angepassten Funktionsbibliotheken oder dem Betriebssystem erforderlich sind. Jede dieser Änderungen hat Auswirkungen auf die Leistung der TAS. [...]„Planung der Umsetzung der Anpassungen in der TAA“ --> Vgl. TAE-Lehrplan v2.0. Kapitel 8.1.2, Abschnitt Refaktorisierung der TAA, um mit den Änderungen im SUT-Schritt zu halten: „Bei der Erweiterung von Funktionalitäten muss darauf geachtet werden, dass diese nicht ad hoc implementiert werden, sondern analysiert und in der TAA geändert werden.“ „Schrittweise Umsetzung der Anpassungen inklusive Smoketests“ --> Vgl. TAE-Lehrplan v2.0. Kapitel 8.1.2, Abschnitt Identifizieren von Änderungen an den Komponenten der Testumgebung: „Änderungen sollten schrittweise vorgenommen werden [...], damit die Auswirkungen auf die TAS durch eine möglichst geringe Anzahl an auszuführenden Tests bewertet werden kann.“ „Durchführung eines vollständigen Regressionstests“ --> Vgl. TAE-Lehrplan v2.0. Kapitel 8.1.2, Abschnitt Identifizieren von Änderungen an den Komponenten der Testumgebung: „Ein vollständiger Regressionstestlauf ist der letzte Schritt, um zu überprüfen, ob sich die Änderung nicht negativ auf die automatisierten Testskripte ausgewirkt hat. Während der Ausführung dieser Regressionstests können Fehlerwirkungen festgestellt werden.“ „Ggf. Analyse der Grundursache von Fehlerwirkungen“ --> Vgl. TAE-Lehrplan v2.0. Kapitel 8.1.2, Abschnitt Identifizieren von Änderungen an den Komponenten der Testumgebung: „Die Identifizierung der Grundursache dieser Fehlerwirkungen (z. B. durch Testberichterstattung, Testprotokolle und Testdatenanalyse) stellt sicher, dass diese nicht auf die Verbesserung der Testautomatisierung zurückzuführen sind.“
- b) **FALSCH** – Im Vergleich zu Antwort a) fehlt die Planung der Umsetzung innerhalb der TAA. Zudem sollte ein vollständiger Regressionstest durchgeführt werden, diese beinhaltet auch nicht-funktionale Tests. Bei fehlgeschlagenen Tests gibt es erst einmal einen Grund, die Änderungen zurückzusetzen. Ein fehlgeschlagener Test ist nicht zwangsläufig auf eine Fehlerwirkung in der TAS zurückzuführen.
- c) **FALSCH** – Die Umsetzung der Änderungen bzw. Anpassungen sollte schrittweise erfolgen und nicht auf einmal. Die Freigabemeldung der TAS an den Testmanager darf nicht erfolgen, bevor potenzielle Fehlerwirkung analysiert wurden. Zudem ist dieser Schritt auch nicht Bestandteil des Lehrplans.
- d) **FALSCH** – Es fehlt die Analyse der Fehlerwirkungen, die möglicherweise durch den Regressionstest aufgedeckt wurden. Bei fehlgeschlagenen Tests gibt es erst einmal einen Grund, die Änderungen zurückzusetzen. Ein fehlgeschlagener Test ist nicht zwangsläufig auf eine Fehlerwirkung in der TAS zurückzuführen.

Frage 40	TAE-8.1.4	K2	Punkte 1.0
----------	-----------	----	------------

Ordnen Sie die nachfolgenden Aktivitäten den entsprechenden weiteren Einsatzmöglichkeiten von Testautomatisierungslösungen zu:

Aktivität:

1. Implementierung von Testskripten, die über eine separat bereitgestellte REST-API Datensätze bereinigen, die bei der Ausführung einer Systemtestsuite modifiziert wurden.
2. Die Releasemanagerin möchte, dass auch jeder erfolgreiche Testschritt der automatisierten GUI-Testfälle mit einem Screenshot dokumentiert wird, um dem Projektmanagement gegenüber bei Fehlerwirkungen im Feld argumentieren zu können.
3. Nach jeder automatisierten Testfallausführung werden die durch die Testfälle erstellten Verzeichnisstrukturen im Zuge des Teardowns komplett gelöscht.
4. Während des Setups einer automatisierten Testsuite im Automotivebereich wird der Hardware-in-the-Loop-Prüfstand vollständig initialisiert, so dass im Anschluss des Setups die automatisierten Testfälle ausgeführt werden können.

Einsatzmöglichkeiten:

- A. Steuerung der Umgebung
- B. Datenmanipulation
- C. Screenshot- und Videoerstellung
- D. Einrichtung der Umgebung

Wählen Sie EINE Option! (1 aus 4)

a)	1-A, 2-C, 3-D, 4-B	☐
b)	1-C, 2-D, 3-A, 4-B	☐
c)	1-D, 2-C, 3-A, 4-B	☐
d)	1-B, 2-C, 3-A, 4-D	☒

TAE-8.1.4 (K2) Der Lernende kann Möglichkeiten für den Einsatz von Testautomatisierungswerkzeugen zusammenfassen.

Begründung:

Die korrekte Kombination ist 1-B, 2-C, 3-A, 4-D, damit ist Option d) korrekt.

Erläuterung zu 1-B: Die Bereinigung von Daten, die während eines Testfalls bzw. einer Testsuite verändert wurden, kann Teil der Testautomatisierung sein. Insbesondere in repräsentativen Umgebungen (Vorproduktionsumgebung bzw. Produktionsumgebung) ist es oftmals notwendig separate Schnittstellen für diesen Zweck einzuführen, insbesondere dann, wenn es per se keine SUT-Schnittstellen für den Zugriff bzw. die Manipulation von Daten gibt. Vgl. TAE-Lehrplan v2.0, Kapitel 8.1.4, Abschnitt Datenalterung: „Mit Hilfe der Testautomatisierung lassen sich Testdaten in der Testumgebung manipulieren.“

Erläuterung zu 2-C: Die Dokumentation des tatsächlichen Verhaltens der SUT kann in einigen Situationen hilfreich sein, um sich gegenüber Vorwürfen der Projektmanagements oder des Kunden bei Fehlerwirkungen im Feld abzusichern. Vgl. TAE-Lehrplan v2.0, Kapitel 8.1.4, Abschnitt Screenshot- und Videoerstellung.

Erläuterung zu 3-A: Die Bereinigung bzw. Manipulation der Umgebung durch die Testautomatisierung ist ein effektives Mittel, um sicherzustellen, dass nachfolgende Testfälle in einer definierten und funktionsfähigen Umgebung ausgeführt werden können. Dazu gehört u.a. auch die Entfernung von Verzeichnissen und Dateien, die während der Testausführung erzeugt wurden. Vgl. TAE-Lehrplan v2.0, Kapitel 8.1.4, Abschnitt Einrichtung und Steuerung der Umgebung.

Erläuterung zu 4-D: Setup-Funktionen ermöglichen es, dass die Testautomatisierung die Einrichtung der Umgebung vornimmt, so dass die Testfälle in einer definierten Konfiguration der Umgebung ausgeführt werden. Vgl. TAE-Lehrplan v2.0, Kapitel 8.1.4, Abschnitt Einrichtung und Steuerung der Umgebung.

Platz für Ihre Notizen:

(werden bei der Korrektur weder gelesen noch bewertet)

Platz für Ihre Notizen:

(werden bei der Korrektur weder gelesen noch bewertet)