Aufzählungspunkt 3

Inhalt

API-Tests sind eine Art von Software-Tests, bei denen eine API validiert wird, um sicherzustellen, dass sie die Erwartungen an Funktionalität, Zuverlässigkeit, Leistung und Sicherheit erfüllt. Es konzentriert sich in erster Linie auf die Business-Logik-Ebene und den Datenaustausch zwischen Softwaresystemen, unabhängig von einer Benutzeroberfläche (UI)

Mit diesem Befehl werden URLs zwischen Texten getestet.

https://policycentral.cloudapps.cisco.com/cppc/policy-advisor/policies/view-policy/1624

Der Verhaltenskodex von Cisco spiegelt wider, wie wir mit Integrität arbeiten und Entscheidungen treffen. Darüber hinaus stellt sie Ressourcen zur Verfügung, mit denen sich komplexe Probleme, wie die verantwortungsvolle Nutzung künstlicher Intelligenz und Interessenkonflikte, bewältigen lassen.

```
function reverseString(str) {
  return str.split("").reverse().join("");
}
```

https://cisco.account.box.com/login

https://cisco.service-now.com/helpzone?id=sc_cat_item&sys_id=a9860b89dbd9a640cb5772fc0f96191d&u_business_service=

Bitten Sie um Unterstützung bei einem aktuellen Problem. Es wird ein Incident Record erstellt und bis zur erfolgreichen Behebung verwaltet. Sie werden ebenfalls über den Fortschritt informiert.

https://www.geeksforgeeks.org/software-testing/software-testing-manual-testing/

Bei der Black Box-Testtechnik überprüft der Tester oder der QA-Analyst nur die Funktionalität des jeweiligen Moduls oder einer bestimmten Methode oder manchmal der gesamten Anwendung, indem er die verschiedenen Testfälle manuell bereitstellt. Hier gibt der Tester die Eingabe für die Anwendung ab und testet sie manuell.

Wenn die erwartete Ausgabe zurückgegeben wird, fährt der Tester mit einer anderen Gruppe von Eingaben fort und meldet dem Team alle Ergebnisse. Wenn die manuelle Eingabe des Benutzers während des Tests fehlschlägt, meldet er dieses Problem an das Entwicklungsteam.

VIDEO TESTEN

Überprüfen	Tabelle
	LINK überprüfen

PRÜFTABELLE

https://cisco.service-now.com/now/sow/record/incident/507c393193e672502c66ff60ed03d632

White Box-Tests

In White Box Testing Technik, die Person wird die interne Struktur des Systems wie Designs überprüfen, Codierung, etc., manuell. Hier überprüft das Entwicklungsteam die gesamte Codierung zeilenweise, um die Richtigkeit des Codes sicherzustellen.

Findet er Unähnlichkeiten oder Fehler im Code, werden diese die Fehler in der Codierung oder den Designs korrigieren oder beheben. Hierbei wird der Prozess vollständig manuell durchgeführt und der Prozess ist effizient, da der Prüfcode bzw. -entwurf manuell von Menschen überprüft wird.

https://en.wikipedia.org/wiki/Manual_testing

https://www.google.com/

Die Prüfung der "bdb developer role" wurde von der ART API zur Entra ID in One Access migriert. Wenn Sie Zugriff anfordern, stellen Sie sicher, dass Sie "Integrationsmethode: memberOf", da es zwei Berechtigungen mit demselben Namen gibt.

Schlüsselaspekte von API-Tests ☐

- Kommunikation auf Nachrichtenebene: Anstelle einer grafischen Benutzeroberfläche interagieren API-Tests direkt mit den Endpunkten (URIs) der Anwendung. Hierzu werden verschiedene HTTP-Methoden (GET, POST, PUT, DELETE) und Datenformate wie JSON oder XML verwendet.
- Frühzeitige Fehlererkennung: API-Tests können bereits in der Entwicklungsphase der Software durchgeführt werden, noch bevor die Benutzeroberfläche erstellt wurde. So können Teams Probleme effizienter und kostengünstiger finden und beheben.
- Automatisierung: Aufgrund der Art der direkten Interaktion und der konsistenten Struktur sind API-Tests für die Automatisierung gut geeignet, was in modernen Agile- und DevOps-

Umgebungen für kontinuierliche Tests in CI/CD-Pipelines von entscheidender Bedeutung ist.
- Umfassende Abdeckung: Sie bietet eine breitere Testabdeckung im Vergleich zum alleinigen Testen der Benutzeroberfläche, einschließlich Testen von Edge-Fällen, Fehlerbehandlung und Sicherheitsschwachstellen, auf die über die Benutzeroberfläche möglicherweise nur schwer zugegriffen werden kann.

Arten von API-Tests ☐
Verschiedene Typen von API-Tests werden verwendet, um verschiedene Aspekte der Qualität einer Anwendung abzudecken: ☐

Katalon

- Funktionstests: Überprüft, ob die API die beabsichtigten Vorgänge ordnungsgemäß durchführt und Eingaben, Ausgaben und Statuscodes wie angegeben verarbeitet.
- Leistungstests: Bewertet die Geschwindigkeit, Stabilität und Skalierbarkeit der API unter verschiedenen Lastbedingungen (z. B. Spitzenlast, Stress).
- Sicherheitstests: Identifiziert Schwachstellen wie SQL Injection, Cross-Site Scripting (XSS) und unterbrochene Authentifizierung/Autorisierung, um vertrauliche Daten zu schützen.
- Integrationstests: Bestätigt, dass verschiedene Teile eines Systems oder externe Dienste, mit denen die API interagiert, nahtlos zusammenarbeiten.
- Vertragstests: Stellt sicher, dass die API einem vereinbarten Vertrag entspricht (Spezifikation wie OpenAPI/Swagger oder WSDL), sodass Unterbrechungen zwischen Service-Updates vermieden werden.
- End-to-End-Tests: Validiert den gesamten Benutzerweg, der mehrere verkettete API-Aufrufe umfasst.
- Arten von API-Tests ☐
  Verschiedene Typen von API-Tests werden verwendet, um verschiedene Aspekte der Qualität einer Anwendung abzudecken: ☐

  Manuelle Tests beginnen mit dem Verständnis der erwarteten Aktionen der Software.

  - Funktionale Anforderungen: Überprüfen Sie Funktionen wie die korrekte Benutzeranmeldung.
  - Nicht funktionsbezogene Anforderungen: Validierung von Leistung, Benutzerfreundlichkeit und Sicherheit (z. B. Ladezeit von Seiten unter 2 Sekunden)
  - Anwenderberichte und Design-Dokumente: Verstehen von Benutzerinteraktionen und Workflows
  - Kommentare von Interessengruppen: Klärung der Anforderungen bei Kunden, Produktmanagern oder Designern

  ## Phase 2: Erstellen eines Testplans

  Ein Testplan definiert die Teststrategie und -ziele.

  - Umfang:Identifiziert zu testende Funktionen und Ausnahmen.
  - Ziele: Stellt die Kernfunktionen und das Anwendererlebnis sicher.
  - Ressourcen: Gibt Teammitglieder, Tools und Zeitpläne an.

- Testverfahren: Umfasst Funktionen, Benutzerfreundlichkeit und Sondierungstests.
- Umgebungen: Definiert Staging oder produktionsähnliche Konfigurationen.

Schritt 3: Design-Testfälle

Testfälle sind klare, schrittweise Skripte, die gründliche manuelle Tests gewährleisten. Testfälle dienen als detaillierte Anleitungen für Tester und stellen sicher, dass jedes Szenario überprüft wird. Jeder Testfall umfasst:

- Test-ID: Ein einzigartiger Code, wie TC_001, für einfache Nachverfolgung.
- Beschreibung: Das Ziel, z. B. die Überprüfung mit gültigen Eingaben.
- Voraussetzungen: Was Sie vor dem Start benötigen, ist z. B. eine Suchseite.
- Schritte: Mögliche Aktionen, z. B. Auswählen des morgigen Datums und Klicken auf "Suchen"
- Erwartetes Ergebnis: Das gewünschte Ergebnis, wie eine Liste der Flüge sortiert nach Preis.
- Nachbedingungen: Das System zeigt die Ergebnisseite an.

Weitere Informationen: Wie werden Testfälle geschrieben?

Schritt 4: Einrichten der Testumgebung

Die Testumgebung sollte der Produktion sehr ähnlich sein.

- Installieren der erforderlichen Anwendungen
- projektspezifische Einstellungen konfigurieren.
- Gewährleistung der Verfügbarkeit von Testdaten
- Überprüfen der Hardware- und Softwareanforderungen

Schritt 5: Durchführung von Testfällen

Führen Sie die Testfälle Schritt für Schritt aus und interagieren Sie als Benutzer mit der Anwendung.

- Tatsächliche Ergebnisse: Was passiert während der Ausführung?
- Erfolgreich/Fehlgeschlagen: Ob das tatsächliche Ergebnis mit dem erwarteten Ergebnis übereinstimmt.
- Beobachtungen: Unerwartetes Verhalten oder Probleme mit der Benutzerfreundlichkeit.

Schritt 6: Fehler protokollieren und melden

Wenn ein Test fehlschlägt oder ein unerwartetes Verhalten auftritt, protokollieren Sie Fehler mit:

- Defekt-ID: Eindeutige Kennung
- Zusammenfassung: Kurzbeschreibung des tatsächlichen Mangels
- Schritte zur Reproduktion: Detaillierte Schritte zum Auslösen des Problems.

- Tatsächliche vs. erwartete Ergebnisse:Was geschah und was hätte geschehen sollen.
- Schweregrad: Überprüfen Sie, ob die Auswirkungen kritisch, schwerwiegend oder gering sind.
- Anhänge:Screenshots, Protokolle oder Videos zum Nachweis des Fehlers.

Schritt 7: Verfolgung und Überprüfung von Defekten

Nach der Anwendung der Korrekturen:

- Verfolgung des Defektstatus im Tool
- Testen Sie die behobenen Probleme erneut.
- Schließen oder erneutes Öffnen von Fehlern anhand der Ergebnisse

Schritt 8: Durchführung von Regressionstests

Durch Regressionstests wird sichergestellt, dass durch Fehlerbehebungen oder neue Änderungen die bestehende Funktionalität nicht beeinträchtigt wird.

- Betroffene Bereiche werden nach der Fehlerbehebung überprüft.
- Überprüfen Sie wichtige Funktionen.
- Überprüfen Sie die Integrationspunkte, um sicherzustellen, dass sie wie gehabt funktionieren.

Schritt 9: Vorbereiten von Testabschlussberichten

Nach Abschluss der Tests die Ergebnisse anhand der Ziele des Testplans berechnen und einen Abschlussbericht für denselben Test erstellen:

- Zusammenfassung: Überblick über die Testaktivitäten
- Testergebnisse: Anzahl der ausgeführten, begangenen und fehlgeschlagenen Testfälle
- Entdeckte Fehler: Gesamtzahl der Fehler, Schweregrad und Lösungsstatus
- Offene Probleme: Alle ungelösten Mängel oder Risiken.
- Erkenntnisse: Erkenntnisse für zukünftige Tests.

Phase 10: Feedback und Empfehlungen

Analysieren Sie die Testergebnisse, um den Interessengruppen aussagekräftiges Feedback zu geben, z. B.:

- Softwarequalität:
- Prozessverbesserungen.
- Künftige Teststrategien
- Einblicke in die Benutzerumgebung.

Für manuelle Tests verwendete Tools

- TestRail: Ein benutzerfreundliches Tool zur Testverwaltung, mit dem manuelle

Testfälle organisiert, ausgeführt und gemeldet werden können und das sich durch eine leistungsstarke Integration und leistungsstarke Dashboards auszeichnet.
- Xray (für Jira): Ein Jira-basiertes Testmanagement-Tool, das manuelle, automatisierte und BDD-Tests mit vollständiger Nachverfolgbarkeit und nahtloser Integration unterstützt
- Qase: Eine moderne Cloud-basierte Testmanagementplattform mit einer einfachen Benutzeroberfläche, der Erstellung von Testfällen mit KI-Unterstützung und integrierter Fehlerverfolgung
- Zephyr: Eine skalierbare Testmanagement-Lösung, die manuelle und explorative Tests mit leistungsstarken Jira-Integrations- und Reporting-Funktionen unterstützt
- Tuskr: Ein leichtes und erschwingliches Cloud-basiertes Testmanagement-Tool mit einer intuitiven Benutzeroberfläche und Collaboration-Funktionen.

Manuelle Tests erforderlich

- Fehlerfrei und stabil:Das Hauptziel manueller Tests besteht darin, sicherzustellen, dass die Anwendung fehlerfrei und stabil ist, den Anforderungen entspricht und den Kunden ein stabiles Produkt liefert.
- Vertrautheit mit dem Produkt: Manuelle Tests helfen den Technikern, sich mit dem Produkt vertraut zu machen und eine Endbenutzerperspektive zu erhalten. Dies hilft ihnen, korrekte Testfälle für die Software zu schreiben.
- Behebung der Mängel: Manuelle Tests helfen sicherzustellen, dass die Fehler vom Entwickler behoben werden und dass die behobenen Mängel erneut geprüft wurden.

Vorteile der manuellen Prüfung

- Schnelles und präzises visuelles Feedback:Es erkennt fast jeden Fehler in der Softwareanwendung und wird verwendet, um die sich dynamisch ändernden GUI-Designs wie Layout, Text usw. zu testen.
- Kostengünstiger: Es ist weniger kostspielig, da es weder besondere Kenntnisse noch einen bestimmten Tooltyp erfordert.
- Keine Codierung erforderlich: Bei Verwendung der Black Box-Testmethode sind keine Programmierkenntnisse erforderlich. Für die neuen Tester ist es leicht zu lernen.
- Effizient bei ungeplanten Änderungen: Bei ungeplanten Änderungen der Applikation ist eine manuelle Prüfung sinnvoll, da diese einfach übernommen werden kann.

Katalon

- Funktionstests: Überprüft, ob die API die beabsichtigten Vorgänge ordnungsgemäß durchführt und Eingaben, Ausgaben und Statuscodes wie angegeben verarbeitet.
- Leistungstests: Bewertet die Geschwindigkeit, Stabilität und Skalierbarkeit der API unter verschiedenen Lastbedingungen (z. B. Spitzenlast, Stress).
- Sicherheitstests: Identifiziert Schwachstellen wie SQL Injection, Cross-Site Scripting (XSS) und unterbrochene Authentifizierung/Autorisierung, um vertrauliche Daten zu schützen.
- Integrationstests: Bestätigt, dass verschiedene Teile eines Systems oder externe Dienste, mit denen die API interagiert, nahtlos zusammenarbeiten.
- Vertragstests: Stellt sicher, dass die API einem vereinbarten Vertrag entspricht (Spezifikation wie OpenAPI/Swagger oder WSDL), sodass Unterbrechungen zwischen Service-Updates vermieden werden.
- End-to-End-Tests: Validiert den gesamten Benutzerweg, der mehrere verkettete API-Aufrufe umfasst.

- So funktioniert es

  Mit visuellen, codeless-Tools können Sie Tests einfach erstellen, erweitern und organisieren, die über APIs, Web-UIs, Datenbanken, ESBs und sogar MCP-Server, die in mit KI infizierten Systemen üblich sind, hinausgehen. Es sind keine tief greifenden technischen Kenntnisse erforderlich. Mit Unterstützung von mehr als 120 Protokollen und Nachrichtenformaten bietet SOAtest ein einheitliches Framework zur End-to-End-Validierung der Geschäftslogik.

  MitSOAtest können Sie:

  - Erstellen Sie szenariobasierte Flows, die reale Geschäftstransaktionen nachahmen und so helfen, versteckte Fehler zu finden, die durch bestimmte Sequenzen ausgelöst werden.
  - Erstellen Sie Testlogik, komplexe Assertionen, Schleifen und datengesteuerte Datenflüsse mit minimalem technischem Fachwissen.
  - Führen Sie einzelne Tests oder vollständige Suites aus, und fügen Sie Regressionssteuerelemente an, um unerwartete Änderungen sofort zu erkennen.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

```
Multiple statements on one line are allowed.
```

Was versteht man unter manuellen Softwaretests?

Die manuelle Prüfung ist das Verfahren, um die Software mithilfe ihrer verschiedenen Funktionen und Merkmale zu überprüfen. Es wird von einer vorgefertigten Reihe von Tests geleitet, die die Software validieren, und liefert einen abschließenden Ergebnisbericht. Diese Art von Tests braucht Zeit für den Abschluss, da sie vollständig durch den manuellen Aufwand durchgeführt wird. Daher liegt bei dieser Art von Tests immer ein menschliches Fehlerpotenzial vor.

Jede neue Software wird zunächst manuell getestet, bevor sie automatisiert wird. Die manuelle Überprüfung einer vollständigen Software nimmt mehr Zeit in Anspruch. Sobald alle Funktionen der Software stabil sind und einwandfrei funktionieren, können einige der manuellen Testfälle in Automatisierung umgewandelt werden. Zunächst werden die manuellen Testfälle ausgewertet, um festzustellen, ob sie vollständig automatisiert werden können. Für diese Art von Tests müssen keine Automatisierungstools verwendet werden, um den gesamten Prozess abzuschließen.

Merkmale der manuellen Softwaretests

Die Merkmale der manuellen Softwaretests sind unten aufgeführt −

- Die manuelle Prüfung erfolgt vollständig mithilfe des menschlichen Eingriffs.
- Die Sondierungsprüfung ist ein wichtiger Teil der manuellen Prüfung. In der Sondierungsprüfung überprüfen die Tester die Software ohne vorbestimmten Testsatz. Es erkennt unvorhergesehene Fehler und verbessert die Kundenzufriedenheit.
- Die manuelle Prüfung ist flexibel, da sie eine Anpassung der Testfälle anhand der geänderten Anforderungen und anderer Testbedingungen ermöglicht.
- Die manuellen Tests können bereits in der Anfangsphase des Software Development Lifecycle (SDLC) durchgeführt werden.
- Einige der komplizierten Testfälle können ohne Automatisierung nur manuell ausgeführt werden.
- Die manuellen Tests sind nützlich, um die Benutzeroberfläche der Software zu validieren. Es hilft, die Anzeige, Reaktionsfähigkeit und das normale Design der Software zu überprüfen.

Warum ist das manuelle Testen der Software erforderlich?

Der manuelle Softwaretest ist aus den unten aufgeführten Gründen erforderlich −

- Die manuellen Tests bestätigen, dass die Software fehlerfrei ist, den Anforderungen entsprechend funktioniert und stabil genug ist, um in der Produktion eingesetzt zu werden.
- Die manuellen Tests ermöglichen es den Testern, sich an die Software zu gewöhnen und zu verstehen, wie die Software mit den Kunden reagiert. Dies hilft, effektive Testfälle zu entwickeln.
- Bei den manuellen Tests werden Fehler in der Software erkannt und behoben.

Schritte des manuellen Softwaretests

Die verschiedenen Schritte des manuellen Tests der Software sind unten aufgeführt −

Schritt 1− Der erste Schritt umfasst die Phase der Anforderungsanalyse, indem die Anforderungs- und Spezifikationsdokumente, Leitfäden usw. durchlaufen werden.

Schritt 2− Der zweite Schritt beinhaltet die Erstellung eines Testplans, der alle Anforderungen berührt.

Schritt 3− Der dritte Schritt beinhaltet die Erstellung von Testfällen, die alle Anforderungen abdecken.

Schritt 4− Der vierte Schritt beinhaltet die Durchführung von Testfällen in der richtigen Testumgebung.

Schritt 5− Der fünfte Schritt umfasst die Analyse der Testergebnisse und die Meldung der Abweichungen als Fehler.

Schritt 6− Der sechste Schritt beinhaltet die Behebung des Defekts und eine erneute Prüfung. Dazu gehört auch die Wiederholung der fehlgeschlagenen Testfälle.

Arten von manuellen Softwaretests

Die verschiedenen Typen von manuellen Softwaretests sind unten aufgeführt −

- Black Box Testing− Es ist die Testtechnik, bei der der Tester keine Kenntnisse über die interne Arbeitsweise der Software hat. Dabei geht es vor allem um die Überprüfung, ob Features und Funktionalitäten entsprechend den Benutzeranforderungen funktionieren.
- White Box Testing− Es ist das Testverfahren, das die Überprüfung der internen Struktur und des Programm-Quellcodes der Software beinhaltet.
- Grey Box Testing− Es ist die Testtechnik, die sowohl die Prinzipien der Black Box verwendet, und White Box Testtechniken.

Tools für manuelle Softwaretests

Nachfolgend sind die verschiedenen Tools aufgeführt, die zum manuellen Testen der Software verwendet werden−

- Testverbindung
- Bugzilla
- Jira
- LoadRunner
- Apache JMeter
- Perfecto

Unterschiede zwischen Softwarehandbuch und Automatisierungstests

Hier ein Vergleich von manuellen und Automatisierungstests der Software −

Manuelle Tests	Automatisierungstests
Es ist das Verfahren, um die Software mit manuellem Aufwand zu überprüfen.	Es ist das Verfahren, die Software mithilfe der Automatisierungstools zu überprüfen.
Es beinhaltet die manuelle Durchführung der Testfälle.	Es beinhaltet die Ausführung der Testfälle mittels Automatisierungsskripten und Tools.
Sie ist weniger produktiv und erfordert mehr Zeit für ihre Fertigstellung.	Sie ist produktiver und erfordert weniger Zeit für ihre Fertigstellung.
Es gewährleistet keine hundertprozentige Testabdeckung.	Es gewährleistet eine größere Testabdeckung als die manuellen Tests.
Es erfordert keine Programmierkenntnisse. Es kann nur mit dem Wissen der Software durchgeführt werden.	Es erfordert Programmierkenntnisse.

Vorteile der manuellen Softwaretests

Die Vorteile von manuellen Softwaretests sind im Folgenden aufgeführt −

- Die manuellen Tests helfen dabei, die sich dynamisch ändernden Elemente auf dem Bildschirm zu überprüfen.
- Die manuelle Prüfung ist billig und nicht auf qualifizierte Ressourcen angewiesen.
- Die manuelle Prüfung kann von Testern ohne Programmierkenntnisse durchgeführt werden.
- Die manuellen Tests können sehr schnell durchgeführt werden und sind geeignet, um unvorhersehbaren Änderungen in der Software Rechnung zu tragen.

Nachteile der manuellen Softwaretests

Die Nachteile von manuellen Softwaretests sind im Folgenden aufgeführt −

- Die manuelle Prüfung ist nicht sehr zuverlässig und bietet Raum für menschliche Fehler.
- Für verschiedene Module müssen separate manuelle Testfälle entwickelt werden, um die Wiederverwendbarkeit zu reduzieren.
- Das manuelle Testen hängt vollständig von der manuellen Ausführung der Tests ab. Einige der Testschritte können jedoch nicht mit dem manuellen Aufwand durchgeführt werden.
- Die Tester, die manuelle Tests durchführen, sollten über die Erfahrung verfügen, mit der Software zu arbeiten. Außerdem gibt es keine Garantie, dass alle Funktionen der Software während der Durchführung der manuellen Tests abgedeckt wurden.
- Das manuelle Testen ist meist eine zeitaufwendige Tätigkeit.

Schlussfolgerung

Damit ist unsere umfassende Einführung in das Tutorial zu manuellen Softwaretests abgeschlossen. Wir begannen mit der Beschreibung, was ist Software manuelle Tests, was sind die Eigenschaften der Software manuelle Tests, warum ist die Software manuelle Tests erforderlich, was sind die verschiedenen Schritte der Software manuelle Tests, was sind die verschiedenen Arten von Software manuelle Tests, was sind die verschiedenen Werkzeuge für Software manuelle Tests verwendet, was sind die Unterschiede zwischen der Software manuelle und Automatisierung Tests, was sind die Vorteile der Software manuelle Tests, und was sind die Nachteile der Software manuelle Tests. Damit verfügen Sie über fundierte Kenntnisse im Bereich der manuellen Softwaretests. Es ist ratsam, das Gelernte weiter zu praktizieren und andere für Softwaretests relevante Bereiche zu erkunden, um Ihr Verständnis zu vertiefen und Ihren Horizont zu erweitern.

Was ist Zugänglichkeit Testen?

Barrierefreiheitstests sind eine Teilmenge von Usability-Tests, bei denen die betreffenden Benutzer Menschen mit allen Fähigkeiten und Behinderungen sind. Die Bedeutung dieses Tests besteht darin, sowohl die Benutzerfreundlichkeit als auch die Zugänglichkeit zu überprüfen.

Barrierefreiheit richtet sich an Personen mit unterschiedlichen Fähigkeiten, wie z. B.:

- Sehbehinderungen
- Körperliche Beeinträchtigung
- Gehörschwäche
- Kognitive Beeinträchtigung
- Lernschwäche

Eine gute Web-Anwendung sollte für alle Personengruppen geeignet sein und NICHT nur für Menschen mit Behinderungen. Dazu gehören:

1. Benutzer mit unzureichender Kommunikationsinfrastruktur
2. Ältere Menschen und neue Nutzer, die oft Computeranalphabeten sind
3. Benutzer, die das alte System verwenden (die neueste Software kann NICHT ausgeführt werden)
4. Benutzer, die nicht standardmäßige Geräte verwenden

5. Benutzer mit eingeschränktem Zugriff

Durchführen von Barrierefreiheitstests

Die Web Accessibility Initiative (WAI) beschreibt die Strategie für Vor- und Konformitätsprüfungen von Websites. Die Web Accessibility Initiative (WAI) umfasst eine Liste von Software-Tools, die bei der Konformitätsbewertung helfen. Diese Tools reichen von speziellen Problemen wie Farbenblindheit bis hin zu Tools, die automatisierte Spider-Tools ausführen.

Web-Barrierefreiheit Testtools

Produkt	Anbieter	URL
ACCverifizieren	HiSoftware	http://www.hisoftware.com
Bobby	WatchFire	http://www.watchfire.com
WebXM	WatchFire	http://www.watchfire.com
Rampe aufsteigen	Deque	http://www.deque.com
Fokus	SSB-Technologien	http://www.ssbtechnologies.com/

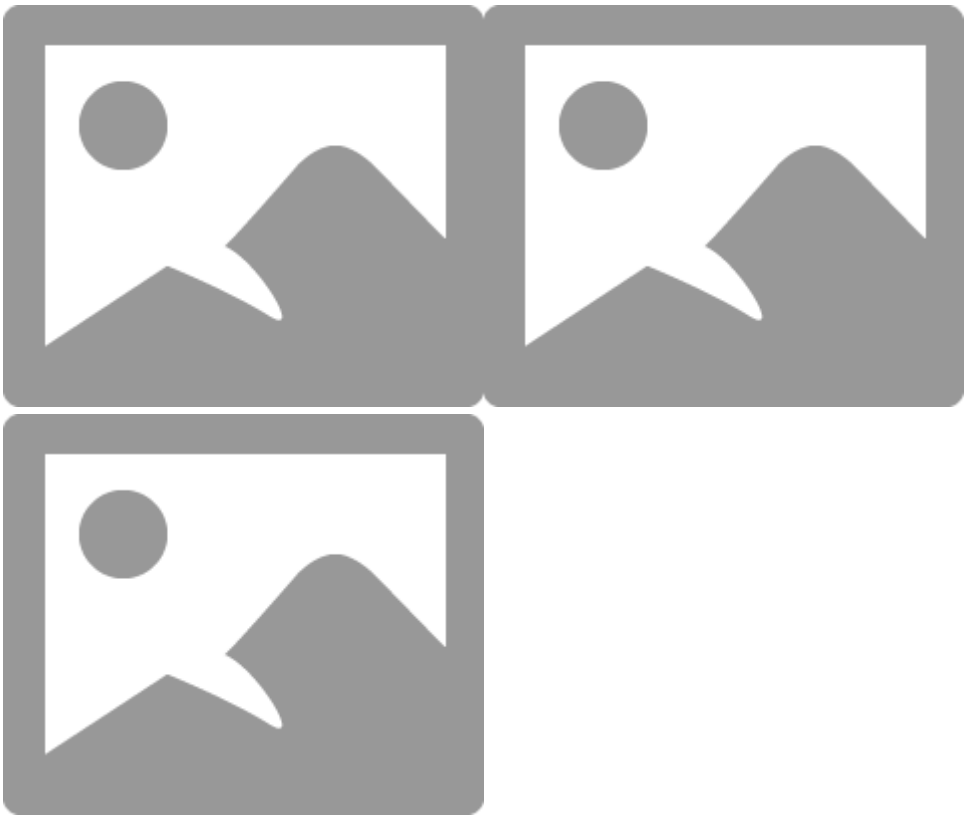
Rolle automatisierter Tools beim Abnahmetest

Die oben genannten automatisierten Zugänglichkeitstestwerkzeuge sind sehr gut bei der Identifizierung von Seiten und Codezeilen, die manuell auf Zugänglichkeit überprüft werden müssen.

1. Überprüfen der Syntax des Standortcodes
2. Suche nach bekannten Mustern, die von Menschen aufgelistet wurden
3. Identifizieren von Seiten mit Elementen, die Probleme verursachen können
4. Identifizierung einiger tatsächlicher Barrierefreiheitsprobleme
5. Identifizierung potenzieller Probleme

Die Interpretation der Ergebnisse aus den automatisierten Zugänglichkeitstesttools erfordert Erfahrung in Zugänglichkeitstechniken mit einem Verständnis von technischen und Usability-

Fragen.



Tests werden sowohl formell als auch informell durchgeführt, um die Softwarequalität zu verbessern. Nach Abschluss der formellen Tests wird eine Runde informeller und willkürlicher Tests durchgeführt. Dies wird als Ad-hoc-Test bezeichnet.

Was ist Ad-hoc-Tests?

Ein Ad-hoc-Test ist eine informelle Testmethode, die auf der Software durchgeführt wird, um Fehler zu finden. Es wird in einem zufälligen Format durchgeführt und wird auch als Affentest bezeichnet. Ad-hoc-Tests folgen keinem systematischen Ansatz und enthalten keine gut dokumentierten Testfälle.

Für Ad-hoc-Tests gibt es keine Dokumentationen, Testszenarien, Testfälle usw. Die Entwickler finden es schwierig, Fehler zu beheben, die durch Ad-hoc-Tests entdeckt wurden, da diese Testdokumente fehlen. Außerdem werden einige kritische, seltene und unerwartete Fehler nur durch zufällige und informelle Tests an der Software identifiziert. Es ist auch eine Art Akzeptanztest und spart die Zeit, neue Testfälle zu erstellen.

Ein praktisches Beispiel für Ad-hoc-Tests ist die Annahme, dass eine Software innerhalb eines Tages an den Kunden ausgeliefert werden muss, und ihre Entwicklung ist nur einen Tag zuvor abgeschlossen. Zu diesem Zeitpunkt bleibt keine Zeit, Testfälle zu erstellen und auszuführen,

sodass das Testteam Ad-hoc-Tests an der gesamten Software auf der Grundlage des allgemeinen Produktwissens und der Erfahrung durchführt.

Ad-hoc-Tests

Die verschiedenen Arten von Ad-hoc-Tests sind im Folgenden aufgeführt −

Buddy-Test

Bei Buddy-Tests sind mindestens zwei Mitglieder während des Testprozesses beteiligt - ein Entwickler und ein Tester. Sobald der Entwickler die Implementierung einer Komponente abgeschlossen hat, führt er Komponententests für diese Komponente durch. Veröffentlichen Sie, dass der Tester einige zufällige, beliebige Daten an die gleiche Komponente sendet und die Ergebnisse untersucht. Im Fehlerfall behebt der Entwickler diese Mängel.

Paartests

Bei der Paarprüfung sind zwei Tester involviert. Einer von ihnen führt eine informelle und zufällige Überprüfung der Software durch, und der andere bewahrt die Testergebnisse auf. So arbeiten beide in einem Paar und tauschen Ideen, Wissen, sodass die Prüfung ordnungsgemäß durchgeführt wird.

Funktionen von Ad-hoc-Tests

Die Funktionen von Ad-hoc-Tests sind im Folgenden aufgeführt −

- Es ist ein zufälliger und informeller Ansatz für Tests.
- Es wird von keiner Dokumentation, von Testszenarien, Fällen usw. unterstützt.
- Sie wird nach Abschluss der formellen Tests durchgeführt.
- Es folgt keinem methodischen oder strukturierten Ansatz.
- Die Durchführung von Ad-hoc-Tests dauert weniger Zeit.
- Es erkennt Fehler in der Software, wenn keine Testfälle verfügbar sind.

Wann werden Ad-hoc-Tests durchgeführt?

Die Ad-hoc-Tests werden in den unten aufgeführten Szenarien durchgeführt &minu;

- Für das Testen der Software steht nur eine begrenzte Zeit zur Verfügung.
- Die formellen Tests wurden abgeschlossen.
- Testfälle sind nicht verfügbar.

Wann werden Ad-hoc-Tests nicht durchgeführt?

Die Ad-hoc-Tests werden in den unten aufgeführten Szenarien nicht durchgeführt −

- Dies geschieht nicht, wenn Fehler durch Ausführen der Testfälle erkannt werden.
- Zum Zeitpunkt der Beta-Tests ist es nicht getan.

Vorteile von Ad-hoc-Tests

Die Vorteile von Ad-hoc-Tests sind im Folgenden aufgeführt −

- Es unterliegt keinem Prozess, sodass Ad-hoc-Tests an jedem Punkt des Software-Entwicklungslebenszyklus durchgeführt werden können.
- Das Testteam kann die Software verifizieren und Fehler finden, indem es neue Testtechniken anwendet, ohne sich nur auf die Testfälle zu verlassen.
- Ein Entwickler kann Ad-hoc-Tests auf demselben Modul durchführen, das er gerade entwickelt, und seine Codequalität erhöhen.
- Während der formale Testprozess viel Zeit in Anspruch nimmt, können Ad-hoc-Tests in kurzer Zeit durchgeführt werden.
- Es ist keine Dokumentation erforderlich.

Nachteile von Ad-hoc-Tests

Die Nachteile von Ad-hoc-Tests sind im Folgenden aufgeführt −

- Ad-hoc-Tests müssen von Teammitgliedern durchgeführt werden, die über Testerfahrung und fundiertes Wissen über das Produkt verfügen. Unerfahrene Teammitglieder können keine Ad-hoc-Tests durchführen.
- Im Falle eines Fehlers ist es schwierig, diesen zu reproduzieren, da der Ad-hoc-Test nicht

von irgendeiner Planung bestimmt wird.

Best Practices für Ad-hoc-Tests

Die Best Practices für Ad-hoc-Tests sind nachstehend aufgeführt −

- Sammeln Sie alle Informationen zum Produkt.
- Identifizieren Sie die defektanfälligen Softwarekomponenten, und priorisieren Sie sie.
- Verwendung geeigneter Testwerkzeuge.

Schlussfolgerung

Damit ist unser umfassendes Tutorial zu Software-Ad-Hoc-Tests abgeschlossen. Wir begannen mit der Beschreibung, was Ad-hoc-Tests sind, welche Arten, Funktionen, Techniken, Vorteile, Nachteile, Zeitpunkte und Best Practices von Ad-hoc-Tests.

Damit verfügen Sie über fundierte Kenntnisse im Software-Ad-hoc-Test. Es ist ratsam, das Gelernte weiter zu praktizieren und andere für Softwaretests relevante Bereiche zu erkunden, um Ihr Verständnis zu vertiefen und Ihren Horizont zu erweitern.

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.