ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Απρίλιος 2020

Επαλήθευση, Επικύρωση & Συντήρηση Λογισμικού

Web Testing

Άγγελος Τσελές

Ανδρέας Πολυχρονάκης

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

Προτού γίνει αναφορά στο Web Testing και εμβαθύνουμε στο θέμα αυτό, είναι απαραίτητο να κατανοήσουμε την ευρύτερη έννοια του Software Testing.

Η Δοκιμή Λογισμικού(Software testing) είναι μια διαδικασία κατά την οποία αξιολογείται η λειτουργικότητα μιας εφαρμογής λογισμικού με στόχο να εξεταστεί εάν το λογισμικό αυτό ικανοποιεί τις πραγματικές απαιτήσεις. Αν δεν τις ικανοποιεί, εντοπίζει τα πιθανά errors ,bugs και τυχόν απαιτήσεις που πρέπει να συμπεριληφθούν.

Το Software Testing είναι εξαιρετικά χρήσιμο και απαραίτητο για την ανάπτυξη ενός προϊόντος λογισμικού, καθώς δίνει την δυνατότητα στον developer να εξετάζει το λογισμικό του σε πραγματικές συνθήκες και έτσι να διασφαλίσει την ποιότητα του.

Υπάρχουν δύο τύποι Software Testing:

1) **Manual Testing**

Ο χειροκίνητος τρόπος ελέγχου επιτρέπει σε έναν developer να επαληθεύσει όλες τις απαραίτητες λειτουργίες του λογισμικού και επίσης να πραγματοποιήσει ελέγχους και να τεστάρει το λογισμικό από την σκοπιά του τελικού χρήστη.

2) **Automation Testing**

Ο αυτοματοποιημένος τρόπος ελέγχου είναι ο βέλτιστος τρόπος για τον εντοπισμό αποκλίσεων μεταξύ των λειτουργιών του σε πραγματικό χρόνο και των απαιτήσεων του.

Ο tester εκτελεί ειδικά scripts ελέγχου από τα οποία παράγονται αυτόματα αποτελέσματα και τα οποία αξιολογούνται.

Οι δύο μέθοδοι Software Testing είναι οι εξής:

1) **Static Testing(ή Επαλήθευση*):*** Η επαλήθευση είναι μία στατική μέθοδος ελέγχου εγγράφων και αρχείων. Αποτελεί την διαδικασία κατά την οποία επαληθεύονται οι απαιτήσεις που έχει ο developer και επαληθεύεται εάν τελικά η ανάπτυξη του λογισμικού εξελίσσεται σωστά.

2) **Dynamic Testing(ή Επικύρωση):** Η επικύρωση είναι μία δυναμική μέθοδος ελέγχου του τελικού λογισμικού. Αποτελεί την διαδικασία κατά την οποία επικυρώνεται εάν το λογισμικό που έχει κατασκευαστεί είναι σωστό ή όχι.

2. Τι είναι Web Testing

A) Ορισμός

Ένας οργανισμός ή μια επιχείρηση διαθέτει μία ιστοσελίδα. Η ιστοσελίδα πρέπει να έχει κάποια χαρακτηριστικά όπως για παράδειγμα να είναι φιλική προς τον χρήστη. Για να διατηρηθούν όλα αυτά τα χαρακτηριστικά και οι λειτουργίες, η ιστοσελίδα πρέπει να υποβληθεί σε μία σειρά από ελέγχους.

Η διαδικασία αυτή ονομάζεται Web Testing. Με απλά λόγια, το Web Testing είναι πρακτική του Software Testing(στο οποίο έγινε ενδεικτική αναφορά προηγουμένως) που ελέγχει εφαρμογές διαδικτύου(web application) ή ιστοσελίδες με στόχο των εντοπισμό πιθανών bug.

Το σύστημα(web-based system) πρέπει να εξεταστεί ολόκληρο προτού γίνει διαθέσιμο για τους τελικούς χρήστες. Με τον τρόπο αυτό εξασφαλίζεται ότι το σύστημα λειτουργεί ομαλά και οι πραγματικοί(real-time) χρήστες έχουν πρόσβαση σε αυτό.

B) Γιατί το Web Testing είναι σημαντικό

Γενικότερα, η φάση του ελέγχου(testing) αποτελεί ένα από τα κρισιμότερα στάδια στην ανάπτυξη λογισμικού ή μιας εφαρμογής λογισμικού .Παρόλα αυτά, ορισμένοι προγραμματιστές υποτιμούν την αξία και την σημασία της. Το πρόβλημα είναι ότι όσο ο προγραμματιστής συνεχίζει να γράφει κώδικα για το λογισμικό αυτό ,οι πιθανότητες εμφάνισης ενός bug είναι ολοένα και περισσότερες και το κόστος επιδιόρθωσης ανεβαίνει με εκθετικό ρυθμό.

Σύμφωνα με έρευνα της IBM, το κόστος επιδιόρθωσης ενός λάθους μετά την έκδοση του προιόντος είναι μέχρι και 4 ως 5 φορές παραπάνω από ότι στο στάδιο σχεδιασμού και ως 100 φορές παραπάνω στο στάδιο της συντήρησης(maintenance).

Επομένως, γίνεται κατανοητό ότι επιθυμητός στόχος είναι ο εντοπισμός ενός λάθους(error) να γίνεται όσο το δυνατόν νωρίτερα και το λογισμικό πρέπει να ελέγχεται εξονυχιστικά πριν γίνει διαθέσιμο στην αγορά. Σε αυτό το πρόβλημα έρχεται να δώσει λύση το Web Testing, το οποίο αποτελείται από πολλά επίπεδα ελέγχου και εξασφαλίζει ότι η εφαρμογή(ή ιστοσελίδα) είναι πλήρως λειτουργική και μπορεί να εκδοθεί.

3. Functionality Testing

A)Τι είναι λειτουργικός έλεγχος

Λειτουργικός Έλεγχος(Functional Testing) ονομάζουμε τον τύπο ελέγχου λογισμικού(Software Testing) ο οποίος επικυρώνει το λογισμικό σύστημα σχετικά με τις λειτουργικές απαιτήσεις.

Σκοπός του Functional Testing είναι να ελέγξει κάθε λειτουργία της εφαρμογής λογισμικού, τοποθετώντας κατάλληλη είσοδο(input) και επαληθεύει το τελικό αποτέλεσμα(output) σε σύγκριση με τις λειτουργικές απαιτήσεις.

Ο λειτουργικός έλεγχος περιλαμβάνει κυρίως black box έλεγχο και δεν ασχολείται με τον πηγαίο κώδικα της εφαρμογής. Ο έλεγχος αυτός ασχολείται με τις λειτουργίες Διεπαφής Χρήστη(User Interface), APIs , Βάσης Δεδομένων, Ασφάλειας και επικοινωνία Πελάτη/Διακομιστή(Client/Server) και μπορεί να γίνει με δύο τρόπους: χειροκίνητα και αυτόματα.

B) Τα κύρια είδη ελέγχου που διενεργούνται

Κύριος στόχος του Functional Testing είναι να γίνει έλεγχος των λειτουργιών του λογισμικού συστήματος.

Επικεντρώνεται στα εξής:

1) **Λειτουργίες κύριας γραμμής(Mainline functions):** Ελέγχονται οι κύριες λειτουργίες της εφαρμογής

2) **Βασική Χρησιμότητα(Basic Usability):** Περιλαμβάνει ελέγχους βασικής χρησιμότητας του συστήματος. Διαπιστώνει εάν ο χρήστης μπορεί ελεύθερα να περιηγηθεί εντός της εφαρμογής/ιστοσελίδας χωρίς δυσκολίες.

3) **Προσβασιμότητα(Accessbility):** Ελέγχει την προσβασιμότητα του συστήματος για τον χρήστη

4) **Καταστάσεις Σφάλματος(Error Conditions):** Γίνεται έλεγχος για εμφάνιση κατάλληλων μηνυμάτων όταν προκύπτουν σφάλματα.

Γ) Πως διενεργείται ο Λειτουργικός Έλεγχος

Για να μπορέσει κάποιος να ελέγξει λειτουργικά μια εφαρμογή, πρέπει να ακολουθήσει τα παρακάτω βήματα:

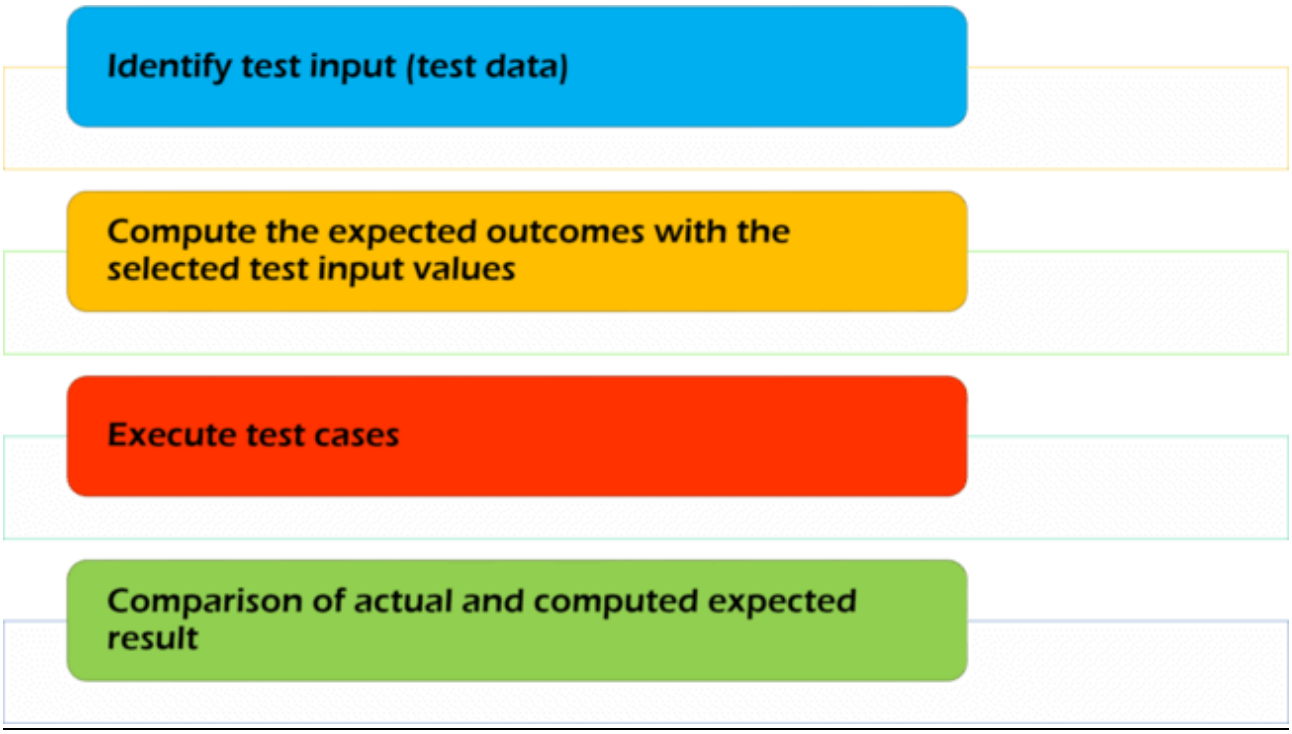
1) Κατανόηση των απαιτήσεων της Τεχνολογίας Λογισμικού

2) Εισαγωγή κατάλληλης εισόδου

3) Υπολογισμός του αναμενόμενου αποτελέσματος με βάση την είσοδο του προηγούμενου βήματος

4) Εκτέλεση περιπτώσεων ελέγχου(test cases)

5) Σύγκριση αναμενόμενου και πραγματικού αποτελέσματος



Εικόνα 1: Τα βασικά βήματα του Functionality Testing

Δ) Λειτουργικός VS Μη Λειτουργικός Έλεγχος

Λειτουργικός Έλεγχος	**Μη Λειτουργικός Έλεγχος**
Ο λειτουργικός έλεγχος πραγματοποιείται χρησιμοποιώντας λειτουργικές απαιτήσεις που έχει δώσει ο πελάτης και ελέγχει το σύστημα με βάση αυτές	Ο μη λειτουργικός έλεγχος ελέγχει την επίδοση, την αξιοπιστία και άλλες μη λειτουργικές απαιτήσεις του συστήματος λογισμικού
Ο λειτουργικός έλεγχος εκτελείται πρώτος	Ο μη λειτουργικός έλεγχος πρέπει να γίνεται μετά τον λειτουργικό
Μπορεί να γίνει είτε χειροκίνητα είτε με την χρήση εργαλείων αυτοματοποίησης	Η χρήση εργαλείων αυτοματοποίησης είναι πολύ πιο αποτελεσματικά
Περιγράφει τι κάνει η εφαρμογή	Περιγράφει το πόσο καλά δουλεύει η εφαρμογή
Ο χειροκίνητος έλεγχος είναι εύκολος	Ο χειροκίνητος έλεγχος είναι δύσκολος
Είσοδο για τον έλεγχο αποτελούν οι λειτουργικές απαιτήσεις του πελάτη	Λαμβάνει ως είσοδο παραμέτρους επίδοσης όπως η ταχύτητα
Παραδείγματα λειτουργικού ελέγχου: Unit Testing, Smoke Testing, Sanity Testing, White box testing, Black Box testing, User Acceptance testing, Regression Testing	Παραδείγματα μη λειτουργικού ελέγχου: Performance Testing, Load Testing, Volume Testing, Stress Testing, Security Testing, Installation Testing, Penetration Testing, Compatibility Testing, Migration Testing

Ε) Ο λειτουργικός έλεγχος στο Web Testing

Περιλαμβάνει:

- Έλεγχος όλων των συνδέσμων(links) της ιστοσελίδας. Εξετάζεται εάν λειτουργούν όλα κανονικά και δεν υπάρχουν broken links.

Τέτοιου είδους σύνδεσμοι είναι:

1) Εξερχόμενοι σύνδεσμοι(Outgoing links)

2) Εσωτερικοί σύνδεσμοι(Internal links)

3) Σύνδεσμοι Anchor

4) Σύνδεσμοι MailTo

- Έλεγχος των φορμών(forms)
- Έλεγχος των cookies. Cookies είναι μικρά αρχεία τα οποία χρησιμοποιούν οι ιστοσελίδες για να θυμούνται τις ενεργές συνεδριάσεις του χρήστη έτσι ώστε να μην χρειάζεται να κάνει αυτός log in κάθε φορά που επισκέπτεται τις ιστοσελίδες αυτές.
- Έλεγχος HTML & CSS κώδικα. Αυτό περιλαμβάνει έλεγχο για συντακτικά λάθη και συμμόρφωση με κάποια standards όπως W3C,OASIS,ISO,ECMA.
- Έλεγχος ροής εργασίας. Εξετάζονται όλα τα πιθανά σενάρια χρήσης όπου ο χρήστης επισκέπτεται όλες τις πιθανές ιστοσελίδες. Παράλληλα εξετάζονται και πιθανές απροσδόκητες ενέργειες όπου θα πρέπει να εμφανίζονται κατάλληλα μηνύματα σφάλματος.

ΣΤ) Εργαλεία Λειτουργικού Ελέγχου

1) RanorexStudio: Μπορεί να εκτελέσει έλεγχο λειτουργικότητας για desktop,web και mobile εφαρμογές

2) Selenium : Γνωστό εργαλείο ελέγχου λειτουργικότητας ανοιχτού κώδικα(open source)

3) QTP: Εργαλείο της HP,αρκετά φιλικό στον χρήστη

4) JUnit: Χρησιμοποιείται κυρίως για εφαρμογές Java

5) soapUI: Είναι εργαλείο ανοιχτού κώδικα και χρησιμοποιείται κυρίως για έλεγχο Web service. Υποστηρίζει πολλά πρωτόκολλα όπως HTTP,SOAP,JDBC

Ζ) Εν κατακλείδι

Ο λειτουργικός έλεγχος είναι μια διαδικασία ελέγχου των λειτουργιών του συστήματος και εξασφαλίζει ότι το σύστημα δουλεύει όπως ακριβώς το έχει ζητήσει η επιχείρηση-πελάτης .Στόχος του ελέγχου αυτού είναι να εξετάζει εάν το σύστημα λειτουργεί στην εντέλεια.

4. Interface Testing

Α) Τι είναι Έλεγχος Διεπαφής

Έλεγχος Διεπαφής(Interface Testing) ορίζεται ως ο τύπος ελέγχου λογισμικού ο οποίος επαληθεύει εάν η επικοινωνία μεταξύ δύο διαφορετικών συστημάτων λογισμικού γίνεται σωστά. Μία σύνδεση που ενσωματώνει δύο λειτουργικές μονάδες ονομάζεται διεπαφή .Η διεπαφή στον χώρο της πληροφορικής μπορεί να έχει πολλές μορφές όπως API, web services κλπ.

Ο έλεγχος αυτών των υπηρεσιών/διεπαφών σύνδεσης αναφέρεται ως Έλεγχος διεπαφής.

Μία διεπαφή ,ουσιαστικά , είναι λογισμικό που αποτελείται από ένα σετ εντολών , μηνυμάτων και άλλων χαρακτηριστικών που καθιστούν ικανή την επικοινωνία μεταξύ μιας συσκευής και ενός χρήστη.

Β) Πως γίνεται ο Έλεγχος Διεπαφής

Ο έλεγχος διεπαφής περιλαμβάνει τον έλεγχο δυο κύριων τμημάτων:

1) Διεπαφή μεταξύ διακομιστή ιστού(Web Server) και διακομιστή εφαρμογής(Application server)

2) Διεπαφή μεταξύ διακομιστή εφαρμογής και διακομιστή βάσης δεδομένων(Database server)

Ο έλεγχος διεπαφής :

- Ελέγχει αν οι διακομιστές εκτελούνται κανονικά
- Χειρίζεται κατάλληλα στα σφάλματα ή επιστρέφει ένα μήνυμα σφάλματος για κάθε επερώτηση που δημιουργεί η εφαρμογή
- Ελέγχει τα αποτελέσματα όταν η σύνδεση σε έναν web server επαναφέρεται

Γ) Παραδείγματα ελέγχου διεπαφής

Έστω για μια εφαρμογή ,η διεπαφή λαμβάνει ως είσοδο ένα αρχείο XML και παραδίδει ως έξοδο ένα αρχείο JSON.Για τον έλεγχο διεπαφής της εφαρμογής το μόνο που χρειάζεται είναι οι προδιαγραφές για αυτούς τους δύο τύπους αρχείων

Με την βοήθεια αυτών των προδιαγραφών μπορούμε να δημιουργήσουμε αρχεία XML τα οποία να στείλουμε ως είσοδο στην διεπαφή .Ο έλεγχος ολοκληρώνεται όταν επαληθευτούν και το αρχείο XML και το αρχείο JSON

Δ) Πλεονεκτήματα Ελέγχου Διεπαφής

- Εξασφαλίζει ότι ο τελικός χρήστης/πελάτης δεν θα αντιμετωπίσει προβλήματα κατά την χρήση του συγκεκριμένου λογισμικού/εφαρμογής
- Εντοπίζει σε ποια πεδία της εφαρμογής έχει πρόσβαση ο τελικός χρήστης και ελέγχει το κατά πόσο αυτά είναι φιλικά προς την χρήση.
- Επαληθεύει απαιτήσεις ασφάλειας κατά την επικοινωνία μεταξύ των συστημάτων
- Ελέγχει εάν μια λύση μπορεί να επιδιορθώσει τυχόν προβλήματα δικτύου μεταξύ του διακομιστή εφαρμογής και της ιστοσελίδας.

Ε) Τύποι Ελέγχου Διεπαφής

1)**Ροή Εργασίας(Workflow*):*** Διασφαλίζει η μηχανή της διεπαφής χειρίζεται την αναμενόμενη ροή εργασίας

2)**Ακραίες Περιπτώσεις/Απροσδόκητες τιμές(Edge cases/unexpected values*):*** Λαμβάνεται υπόψιν όταν ο έλεγχος περιλαμβάνει ημερομηνία ,μήνα και ώρα αντίστροφα

3)**Επίδοση, Φόρτος, Έλεγχος Δικτύου(Perfomance ,load, and network testing):** Μία διεπαφή με αυξημένη κίνηση ίσως απαιτεί μεγαλύτερο έλεγχο φόρτου σε σχέση με μια διεπαφή με χαμηλότερη κίνηση. Αυτό εξαρτάται από την μηχανή της διεπαφής και την υποδομή της συνδεσιμότητας

4)**Ξεχωριστά συστήματα**: Κάθε σύστημα ελέγχεται ξεχωριστά.

ΣΤ) Ο Έλεγχος Διεπαφής στο Web Testing

Ελέγχονται 3 στρώματα(layers).Η εφαρμογή ,ο διακομιστής δικτύου και ο διακομιστής της βάσης δεδομένων

-**Εφαρμογή**: Στέλνονται αιτήσεις προς την βάση δεδομένων και η έξοδος(output) εμφανίζεται κανονικά στην πλευρά του πελάτη. Αν η εφαρμογή εντοπίσει κάποιο λάθος , αυτό θα πρέπει να το δείξει μόνο στον διαχειριστή(administrator) και όχι στον χρήστη.

-**Διακομιστής Ιστού(Web Server)**: Ο διακομιστής ιστού χειρίζεται όλα τα αιτήματα της εφαρμογής χωρίς να αντιμετωπίσει πρόβλημα άρνησης εξυπηρέτησης(service denial)

-**Διακομιστής Βάσης Δεδομένων(Database Server)**: Διασφαλίζει ότι οι επερωτήσεις που θα σταλθούν σε αυτόν θα δώσουν τα αναμενόμενα αποτελέσματα.

Έλεγχος Απόκρισης Συστήματος πραγματοποιείται όταν η σύνδεση μεταξύ των 3 παραπάνω στρωμάτων δεν μπορεί να εγκαθιδρυθεί και το κατάλληλο μήνυμα εμφανίζεται στον τελικό χρήστη.

Η) Έλεγχος Διεπαφής VS Έλεγχος Ενσωμάτωσης

Έλεγχος Διεπαφής: Είναι ένας τύπος ελέγχου ενσωμάτωσης ο οποίος ασχολείται με τον έλεγχο των διεπαφών μεταξύ εξαρτημάτων ή συστημάτων

Έλεγχος Ενσωμάτωσης: Ο έλεγχος γίνεται για τον εντοπισμό αποκλίσεων στις διεπαφές και τις αλληλεπιδράσεις μεταξύ ενσωματωμένων εξαρτημάτων ή συστημάτων

Ζ) Εν κατακλείδι

Ο έλεγχος διεπαφής είναι έλεγχος της σύνδεσης που ενσωματώνει δύο τμήματα ενός συστήματος, το οποίο ονομάζεται διεπαφή

Εξασφαλίζει ότι ο χρήστης δεν πρόκειται να αντιμετωπίσει κάποιο πρόβλημα όσο χρησιμοποιεί το συγκεκριμένο προϊόν λογισμικού.

5. Compatibility Testing

Α) Ορισμός

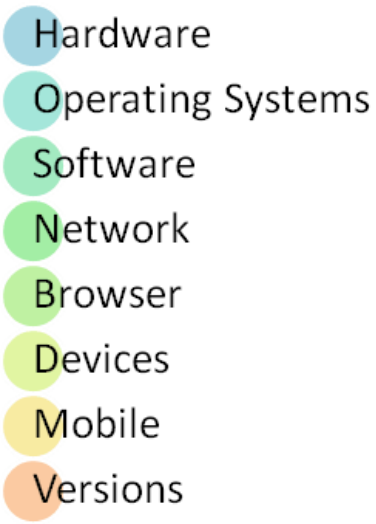
Έλεγχος συμβατότητας(Compatibility Testing) είναι ένας τύπος ελέγχου λογισμικού ο οποίος εξετάζει εάν ένα λογισμικό μπορεί να τρέχει σε διαφορετικά hardware,λειτουργικά συστήματα, εφαρμογές, περιβάλλοντα δικτύου ή κινητές συσκευές. Ανήκει στην κατηγορία του μη λειτουργικού ελέγχου .

Β) Τύποι Ελέγχων συμβατότητας

1)**Hardware**: Ελέγχει εάν το λογισμικό είναι συμβατό με διαφορετικό hardware.

2)**Λειτουργικό Σύστημα(Operating System):** Ελέγχει αν το λογισμικό είναι συμβατό με πολλαπλά λειτουργικά συστήματα όπως Windows,Unix,Mac OS.

3)**Λογισμικό**: Ελέγχει εάν το λογισμικό είναι συμβατό και με άλλα λογισμικά. Για παράδειγμα η εφαρμογή MS Word πρέπει να είναι συμβατή με τις άλλες εφαρμογές Office όπως το Outlook και το Excel.



Εικόνα 2: Οι 7 τύποι ελέγχου συμβατότητας

4)**Δίκτυο**: Γίνεται εκτίμηση της επίδοσης ενός συστήματος σε ένα δίκτυο με παραμέτρους που μπορεί να αλλάζουν από δίκτυο σε δίκτυο. Τέτοιες παράμετροι είναι το εύρος ζώνης(bandwidth) και η χωρητικότητα(Capacity).

5)**Εξυπηρετητής(Browser)**: Ελέγχει εάν η ιστοσελίδα είναι συμβατή με διάφορους εξυπηρετητές όπως Firefox,Google Chrome,Internet Explorer.

6)**Συσκευή**: Ελέγχει εάν το λογισμικό είναι συμβατό με συσκευές όπως συσκευές με θύρες USB, εκτυπωτές και συσκευές bluetooth.

7)**Κινητή Συσκευή**: Ελέγχει εάν το λογισμικό είναι συμβατό με λειτουργικά συστήματα όπως Android και iOS.

8)**Εκδόσεις Λογισμικού**: Ελέγχει εάν το λογισμικό είναι συμβατό με διαφορετικές εκδόσεις ενός άλλου λογισμικού. Για παράδειγμα ελέγχει εάν το MS Word είναι συμβατό με τα Windows 7,Windows 7 SP1,Windows 7 SP2,Windows 7 SP3.

Υπάρχουν δύο τύποι ελέγχου έκδοσης(version checking)

-**Οπισθοδρομικός έλεγχος συμβατότητας(Backward compatibility Testing**): Επαληθεύει το πως συμπεριφέρεται το λογισμικό με παλαιότερες εκδόσεις software/hardware.

-**Προς τα εμπρός έλεγχος συμβατότητας(Forward compatibility Testing):** Επαληθεύει το πως συμπεριφέρεται το λογισμικό με νεότερες εκδόσεις software/hardware.

Γ) Ο έλεγχος συμβατότητας στο Web Testing

Ο έλεγχος συμβατότητας εξασφαλίζει ότι η εφαρμογή ιστού μπορεί να λειτουργήσει το ίδιο ομαλά σε διαφορετικές συσκευές. Εστιάζει κυρίως σε δύο τύπους, τον εξυπηρετητή και το λειτουργικό σύστημα. Γίνονται έλεγχοι και εξετάζονται και οι δύο όπως ακριβώς περιγράψαμε προηγουμένως.

Χρησιμοποιούμε διαφορετικά εργαλεία για τον κάθε τύπο. Για τον εξυπηρετητή, ένα πολύ χρήσιμο είναι το BrowserStack το οποίο βοηθάει έναν μηχανικό λογισμικού να ελέγχει μια εφαρμογή σε διαφορετικούς εξυπηρετητές.

Για το λειτουργικό σύστημα, γίνεται χρήση των Virtual Desktop.Χρησιμοποιείται για να τρέχουν οι εφαρμογές σε πολλαπλά λειτουργικά συστήματα ως εικονικές μηχανές(virtual machines).

Δ) Πώς γίνεται ο έλεγχος συμβατότητας

Βήμα 1ο: Αρχικά πρέπει να οριστούν τα περιβάλλοντα ή οι πλατφόρμες πάνω στις οποίες θέλουμε η εφαρμογή να δουλεύει.

Βήμα 2ο: Ο tester πρέπει να έχει επαρκείς γνώσεις για τα περιβάλλοντα, το λογισμικό και το hardware έτσι ώστε να γνωρίζει εκ των προτέρων και με βεβαιότητα το πως θέλει να συμπεριφέρεται η εφαρμογή ανάλογα τις διαφορετικές διαμορφώσεις(configurations).

Βήμα 3ο: Το περιβάλλον πρέπει να στηθεί κατάλληλα με πολλαπλές πλατφόρμες, συσκευές και δίκτυα και να ελεγχθεί εάν η εφαρμογή λειτουργεί κανονικά.

Βήμα 4ο: Αναφορά προβλημάτων(Report bugs).Επιδιόρθωση τυχόν αποκλίσεων και ξανά έλεγχος συμβατότητας.

Ε) Εν κατακλείδι

Ο έλεγχος συμβατότητας χρησιμοποιείται για να εξετάσουμε αν ένα λογισμικό μπορεί να δουλέψει κάτω από διαφορετικές πλατφόρμες/συστήματα. Αυτός ο έλεγχος είναι απαραίτητος ώστε να εξασφαλιστεί ότι η συγκεκριμένη εφαρμογή είναι συμβατή με το περιβάλλον του πελάτη.

6. Security Testing

A) Ορισμός

Έλεγχο Ασφαλείας(Security Testing) ονομάζουμε τον τύπο ελέγχου λογισμικού ο οποίος εντοπίζει τρωτά σημεία(vulnerabilities),απειλές και πιθανά ρίσκα σε μία εφαρμογή λογισμικού και αποτρέπει κακόβουλες επιθέσεις από εισβολείς.

Σκοπός είναι ο εντοπισμός όλων των πιθανών αδυναμιών του συστήματος λογισμικού το οποίο μπορεί να έχει σαν αποτέλεσμα την απώλεια πληροφορίας , εσόδων και άλλων πολύτιμων στοιχείων για έναν ιδιώτη ή οργανισμό.

Έτσι , πρέπει να βρεθούν όλες οι απειλές του συστήματος και να υπολογιστούν πιθανές αδυναμίες ώστε οι προγραμματιστές να μπορέσουν να επιδιορθώσουν αυτά τα προβλήματα μέσα από τον κώδικα και το σύστημα να μην σταματήσει την λειτουργία του και να μην γίνει στόχος επιθέσεων.

B) Τύποι ελέγχων ασφαλείας

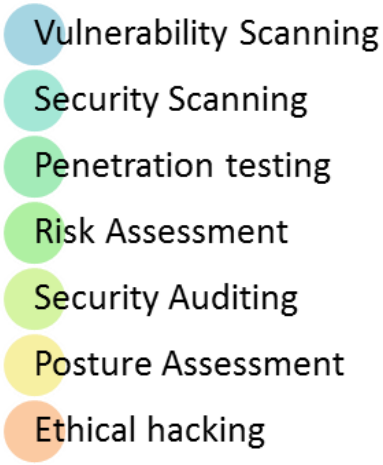
1) **Σάρωση για τρωτά σημεία(Vulnerability Scanning):** Αυτοματοποιημένο λογισμικό σαρώνει ένα σύστημα για γνωστά τρωτά σημεία.

2) **Σάρωση ασφαλείας(Security Scanning):** Περιλαμβάνει την αναγνώριση αδυναμιών του δικτύου και του συστήματος και παρέχει λύσεις για την εξάλειψη αυτών των κινδύνων .Μπορεί να γίνει χειροκίνητα και αυτοματοποιημένα.

3) **Έλεγχος για διείσδυση(Penetration Testing):** Προσομοιώνει μία επίθεση από έναν κακόβουλο χάκερ .Ο έλεγχος αυτός περιλαμβάνει επίσης ανάλυση ενός συγκεκριμένου συστήματος για έλεγχο πιθανών τρωτών σημείων που μπορεί να εκμεταλλευτεί ένας κακόβουλος χάκερ.

4) **Εκτίμηση ρίσκου(Risk Assessment):** Γίνεται ανάλυση για τους κινδύνους ασφαλείας που παρατηρούνται στον οργανισμό. Τα ρίσκα αυτά χαρακτηρίζονται ως χαμηλά ,μέτρια και υψηλά .Τέλος, προτείνονται μέτρα για την μείωση του κινδύνου.

5) **Έλεγχος ασφαλείας(Security Auditing):** Εσωτερική επιθεώρηση των εφαρμογών και των λειτουργικών συστημάτων για τρύπες/ελαττώματα ασφαλείας.



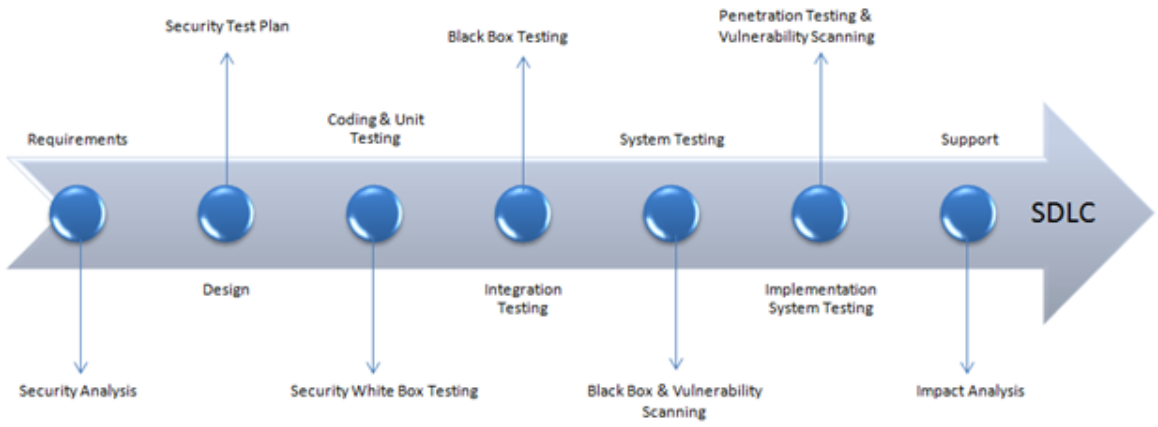
Εικόνα 3: Οι 7 κυριότεροι τύποι ελέγχου ασφαλείας

6) **Ηθική Πειρατεία(Ethical Hacking*):*** Γίνεται επίθεση στο λογισμικό σύστημα ενός οργανισμού με σκοπό να φανερωθούν τυχόν τρύπες στην ασφάλεια του συστήματος.

7) **Εκτίμηση κορμού(Posture Assessment):** Περιλαμβάνει Security scanning, Ethical Hacking και Risk Assessments και έτσι φανερώνεται μια γενική όψη για την ασφάλεια του συστήματος ενός οργανισμού.

Γ) Πως γίνεται ο έλεγχος ασφαλείας

Είναι ευρέως αποδεκτό ότι αν αναβάλουμε τον έλεγχο ασφαλείας μετά την υλοποίηση του λογισμικού, το κόστος θα είναι πολλαπλάσιο. Οπότε εύλογα προκύπτει το συμπέρασμα ότι ο έλεγχος ασφαλείας πρέπει να γίνει στα αρχικότερα στάδια και συγκεκριμένα στον κύκλο ζωής SDLC.



Εικόνα 4: Οι διαδικασίες στον κύκλο ζωής SDLC

Το πλάνο ελέγχου πρέπει να περιλαμβάνει:

- ✓ Τεστ/Σενάρια που να σχετίζονται με την ασφάλεια
- ✓ Δεδομένα για έλεγχο ασφαλείας
- ✓ Κατάλληλα εργαλεία για τον έλεγχο
- ✓ Ανάλυση για το αποτέλεσμα των ελέγχων από διαφορετικά εργαλεία ασφάλειας

Στάδια SDLC	**Διεργασίες Ασφαλείας**
Απαιτήσεις	Ανάλυση ασφαλείας για απαιτήσεις και έλεγχος για περιπτώσεις κατάχρησης/λανθασμένης χρήσης(abuse/misuse cases)
Σχεδιασμός	Ανάλυση ασφαλείας για προβλήματα σχεδιασμού. Ανάπτυξη πλάνου ελέγχου το οποίο περιλαμβάνει και ελέγχους ασφαλείας.
Έλεγχος κώδικα και μονάδας	Στατικός και δυναμικός έλεγχος και έλεγχος ασφαλείας White Box
Έλεγχος Ενσωμάτωσης	Έλεγχος Black Box
Έλεγχος Συστήματος	Έλεγχος Black Box και Σάρωση για τρωτά σημεία
Υλοποίηση	Έλεγχος για διείσδυση, Σάρωση για τρωτά σημεία
Υποστήριξη	Ανάλυση των επιπτώσεων

Δ) Παραδείγματα σεναρίων ελέγχου ασφαλείας

- Ο κωδικός πρέπει να είναι κρυπτογραφημένος
- Η εφαρμογή ή το σύστημα δεν πρέπει να επιτρέπει την είσοδο σε λανθασμένους χρήστες(invalid users)
- Έλεγχος για cookies και χρόνου συνεδρίασης(session time) της εφαρμογής
- Για ιστοσελίδες με οικονομικό περιεχόμενο, το κουμπί Πίσω(back button) του εξυπηρετητή δεν πρέπει να λειτουργεί.

Ε) Τεχνικές/Προσεγγίσεις στον Έλεγχο Ασφαλείας

Στον έλεγχο ασφαλείας χρησιμοποιούνται διάφορες μεθοδολογίες. Μερικές από αυτές είναι:

- **Tiger Box**: Πραγματοποιείται συνήθως σε ένα laptop που διαθέτει πολλά λειτουργικά σύστημα και εργαλεία hacking.
- **Black Box**: Ο tester είναι εξουσιοδοτημένος να ελέγξει οτιδήποτε αφορά την τοπολογία δικτύου.
- **Grey Box**: Ορισμένες πληροφορίες δίνονται στον tester σχετικά με το σύστημα και είναι ένα κράμα μεταξύ των μοντέλων white και black box.

ΣΤ) Ρόλοι στον Έλεγχο Ασφαλείας

- **Χάκερ**: Αποκτούν πρόσβαση σε συστήματα υπολογιστών ή δικτύου χωρίς εξουσιοδότηση
- **Κράκερ**: Εισχωρούν σε συστήματα και κλέβουν ή καταστρέφουν δεδομένα.
- **Ηθικοί χάκερ**: Κάνουν ενέργειες εισχώρησης σε συστήματα αλλά με την έγκριση του ιδιοκτήτη.
- **Σεναριόπαιδα(Script Kiddies):** Άπειροι χάκερ με περιορισμένη γνώση προγραμματιστικών γλωσσών.

Ζ) Ο έλεγχος ασφαλείας στο Web Testing

Ο έλεγχος ασφαλείας είναι κρίσιμος για σελίδες με ηλεκτρονικές πληρωμές καθώς αποθηκεύουν ευαίσθητες πληροφορίες του πελάτη όπως πιστωτικές κάρτες.

Ο έλεγχος περιλαμβάνει:

- Έλεγχος για μη εξουσιοδοτημένη πρόσβαση ώστε να διασφαλιστεί ότι οι σελίδες δεν θα προσπελαύνονται.
- Αρχεία με ευαίσθητο περιεχόμενο δεν μπορούν να κατεβαστούν χωρίς την απαραίτητη άδεια
- Έλεγχος για παρατεταμένη αδράνεια χρήστη και τερματισμός της συνεδρίασης
- Η ιστοσελίδα πρέπει να κατευθύνει σε κρυπτογραφημένες σελίδες κατά πιστοποίηση SSL.

Η) Εργαλεία

- **Owasp**: Είναι ένας μη κερδοσκοπικός οργανισμός ο οποίος στοχεύει στην βελτιστοποίηση της ασφάλειας λογισμικού. Διαθέτει πολλά εργαλεία όπως:

  -Zed Attack Proxy(ZAP): εργαλείο ελέγχου ενσωματωμένης διείσδυσης(integrated penetration)

  -OWASP Dependency Check : σαρώνει όλο το πρότζεκτ και ελέγχει για γνωστά κενά ασφαλείας

  -OWASP Web Testing Environment Project: συλλογή από εργαλεία ασφαλείας

- **WireShark**: Είναι εργαλείο ανάλυσης δικτύου. Συλλέγει ένα πακέτο σε πραγματικό χρόνο και παρέχει λεπτομέρειες σχετικά με αυτό όπως πρωτόκολλο, αποκρυπτογράφηση και άλλες πληροφορίες. Είναι πρόγραμμα ανοιχτού κώδικα και μπορεί να χρησιμοποιηθεί σε συστήματα που τρέχουν Windows ,Linux ,OS X κλπ.

- **W3af**: Είναι μια εφαρμογή ιστού. Έχει τρεις τύπους διαφορετικών plugin: discovery, audit, attack τα οποία συνεργάζονται μεταξύ τους για τον εντοπισμό τρωτών σημείων σε μία ιστοσελίδα.

Ι) Μύθοι και αλήθειες σχετικά με τον έλεγχο ασφαλείας

Μύθος*: Είμαστε μικρή επιχείρηση οπότε δεν χρειαζόμαστε πολιτική ασφαλείας(security policy)*

Αλήθεια: Ο καθένας είτε ως επιχείρηση είτε ως ιδιώτης πρέπει να μεριμνεί για την πολιτική ασφαλείας.

Μύθος: *Η επένδυση στον έλεγχο ασφαλείας δεν έχει κάποιο αντίκρισμα.*

Αλήθεια: Ο έλεγχος ασφαλείας μπορεί να υποδείξει σημεία και τομείς οι οποίοι επιδέχονται βελτίωση και έτσι να βελτιωθεί η αποδοτικότητα και η μέγιστη δυνατή ικανότητα διεκπεραίωσης

Μύθος*: Το διαδίκτυο δεν είναι ασφαλές οπότε θα αγοράσω λογισμικό ή hardware για την μέγιστη προστασία*

Αλήθεια: Η αγορά τέτοιου υλικού δεν αρκεί από μόνη της. Είναι σημαντικό να καταλάβει κανείς πως λειτουργεί η ασφάλεια και να προβεί τότε σε ανάλογες κινήσεις.

Κ) Εν κατακλείδι

Ο έλεγχος ασφαλείας είναι ο πιο σημαντικός έλεγχος για μία εφαρμογή και ελέγχει εάν οι εμπιστευτικές πληροφορίες παραμένουν εμπιστευτικές. Σε αυτού του είδους ελέγχου ο tester διαδραματίζει τον ρόλο του επιτιθέμενου και "οργώνει" το σύστημα με σκοπό τον εντοπισμό πιθανών λαθών ασφαλείας.

Είναι πολύ σημαντικό να συνειδητοποιήσουμε την αξία του ελέγχου ασφαλείας στον τομέα της Τεχνολογίας Λογισμικού και το πως πρέπει αυτή να διασφαλίζεται.

7. Perfomance Testing

Α) Ορισμός

Ο **Έλεγχος Επίδοσης(Performance Testing)** ελέγχει την ταχύτητα, τον χρόνο απόκρισης, την αξιοπιστία, τη χρήση πόρων, την επεκτασιμότητα ενός προγράμματος λογισμικού υπό τον αναμενόμενο φόρτο εργασίας τους. Ο σκοπός του Performance Testing δεν είναι να εντοπίσουμε λειτουργικά ελαττώματα, αλλά να εξαλείψουμε τα εμπόδια απόδοσης στο λογισμικό ή τη συσκευή.

Β) Για ποιο λόγο κάνουμε Performance Testing ?

Ο Έλεγχος απόδοσης γίνεται για να παρέχει στους ενδιαφερόμενους πληροφορίες σχετικά με την εφαρμογή τους σχετικά με την ταχύτητα, τη σταθερότητα και την επεκτασιμότητα. Το πιο σημαντικό, το Performance Testing αποκαλύπτει τι πρέπει να βελτιωθεί προτού το προϊόν κυκλοφορήσει στην αγορά.

Ο έλεγχος απόδοσης θα καθορίσει εάν το λογισμικό τους πληροί τις απαιτήσεις ταχύτητας, κλιμάκωσης και σταθερότητας υπό τον αναμενόμενο φόρτο εργασίας. Οι εφαρμογές που αποστέλλονται στην αγορά με μετρήσεις κακής απόδοσης λόγω ανύπαρκτης ή κακής δοκιμής απόδοσης είναι πιθανό να αποκτήσουν κακή φήμη και να μην επιτύχουν τους αναμενόμενους στόχους πωλήσεων.

Ως εκ τούτου, ο έλεγχος απόδοσης είναι σημαντικός.

Γ) Είδη Ελέγχου Επίδοσης

***1)*Load Testing**

Δοκιμή φόρτωσης - ελέγχει την ικανότητα της εφαρμογής να εκτελεί υπό αναμενόμενα φορτία χρήστη. Ο στόχος είναι ο εντοπισμός σημείων συμφόρησης πριν από την εφαρμογή της εφαρμογής λογισμικού.

***2)*Stress Testing**

Περιλαμβάνει τη δοκιμή μιας εφαρμογής υπό ακραίο φόρτο εργασίας για να δείτε πώς χειρίζεται την υψηλή κυκλοφορία ή την επεξεργασία δεδομένων. Ο στόχος είναι να προσδιοριστεί το σημείο διακοπής μιας εφαρμογής.

***3)*Endurance Testing**

Γίνεται για να βεβαιωθούμε ότι το λογισμικό μπορεί να χειριστεί το αναμενόμενο φορτίο για μεγάλο χρονικό διάστημα.

***4)*Spike Testing**

Ελέγχει την αντίδραση του λογισμικού σε ξαφνικές μεγάλες αυξήσεις στο φορτίο που δημιουργούν οι χρήστες.

***5)*Scalability Testing**

Ο στόχος της δοκιμής κλιμάκωσης είναι να προσδιοριστεί η αποτελεσματικότητα της εφαρμογής λογισμικού στην "κλιμάκωση" για την υποστήριξη της αύξησης του φορτίου του χρήστη. Βοηθά στον προγραμματισμό της προσθήκης χωρητικότητας στο σύστημα λογισμικού.

Δ) Πιθανά προβλήματα που μπορεί να παρουσιαστούν

- **Long load time**: Ο χρόνος φόρτωσης είναι συνήθως ο αρχικός χρόνος για την εκκίνηση μιας εφαρμογής. Αυτό πρέπει γενικά να περιοριστεί στο ελάχιστο. Ενώ ορισμένες εφαρμογές είναι αδύνατο να φορτώσουν σε λιγότερο από ένα λεπτό, ο χρόνος φόρτωσης πρέπει να διατηρείται κάτω από λίγα δευτερόλεπτα εάν είναι δυνατόν.

- **Poor response time**: Ο χρόνος απόκρισης είναι ο χρόνος από τον οποίο ένας χρήστης εισάγει δεδομένα στην εφαρμογή έως ότου η εφαρμογή αποδώσει μια απόκριση σε αυτήν την είσοδο. Γενικά, αυτό πρέπει να είναι πολύ γρήγορο. Και πάλι, εάν ένας χρήστης πρέπει να περιμένει πολύ, χάνει το ενδιαφέρον του.

- **Poor scalability**: Ένα προϊόν λογισμικού υποφέρει από κακή επεκτασιμότητα όταν δεν μπορεί να χειριστεί τον αναμενόμενο αριθμό χρηστών ή όταν δεν εξυπηρετεί αρκετά μεγάλο εύρος χρηστών. Η δοκιμή φορτίου πρέπει να γίνει για να βεβαιωθείτε ότι η εφαρμογή μπορεί να χειριστεί τον αναμενόμενο αριθμό χρηστών.

- **Bottlenecking**: Bottleneck σε ένα σύστημα ονομάζουμε τα εμπόδια που προκαλούν μειωμένη συνολική επίδοση του συστήματος. Προκαλείται συνήθως είτε από λάθη στον κώδικα είτε από προβλήματα hardware.Μπορεί να επιδιορθωθεί είτε με την διόρθωση του τμήματος κώδικα που προκαλεί το πρόβλημα είτε με την αύξηση/βελτίωση του υλικού hardware/

Ε) Η διαδικασία ελέγχου επίδοσης

Ακολουθεί μια γενική διαδικασία για τον τρόπο εκτέλεσης του performance testing

1) Εντοπισμός του περιβάλλοντος που πρόκειται να γίνει ο έλεγχος

Μαθαίνει το φυσικό περιβάλλον δοκιμής, το περιβάλλον παραγωγής και ποια εργαλεία δοκιμών είναι διαθέσιμα. Κατανοεί τις λεπτομέρειες του υλικού, του λογισμικού και των διαμορφώσεων δικτύου που χρησιμοποιήθηκαν κατά τη διάρκεια της δοκιμής πριν ξεκινήσετε τη διαδικασία δοκιμής. Θα βοηθήσει τους δοκιμαστές να δημιουργήσουν πιο αποτελεσματικές δοκιμές. Θα βοηθήσει επίσης στον εντοπισμό πιθανών προκλήσεων που ενδέχεται να αντιμετωπίσουν οι υπεύθυνοι δοκιμών κατά τη διάρκεια των διαδικασιών δοκιμής ***απόδοσης.***

2) **Εντοπισμός των αποδεκτών κριτηρίων επίδοσης**

Αυτό περιλαμβάνει στόχους και περιορισμούς για την απόδοση, τους χρόνους απόκρισης και την κατανομή πόρων. Είναι επίσης απαραίτητο να προσδιοριστούν τα κριτήρια επιτυχίας του έργου εκτός αυτών των στόχων και περιορισμών. Οι υπεύθυνοι δοκιμών θα πρέπει να έχουν την εξουσία να ορίζουν κριτήρια και στόχους απόδοσης, επειδή συχνά οι προδιαγραφές του έργου δεν θα περιλαμβάνουν αρκετά μεγάλη ποικιλία κριτηρίων απόδοσης. Μερικές φορές μπορεί να μην υπάρχει καθόλου. Όταν είναι δυνατόν, η εύρεση μιας παρόμοιας εφαρμογής για σύγκριση είναι ένας καλός τρόπος για να θέσετε στόχους απόδοσης.

3)**Σχεδιασμός των ελέγχων επίδοσης**

Προσδιορίστε τον τρόπο με τον οποίο η χρήση είναι πιθανό να διαφέρει μεταξύ των τελικών χρηστών και προσδιορίστε βασικά σενάρια για δοκιμή για όλες τις πιθανές περιπτώσεις χρήσης. Είναι απαραίτητο να προσομοιώσετε μια ποικιλία τελικών χρηστών, να σχεδιάσετε δεδομένα δοκιμών απόδοσης και να περιγράψετε ποιες μετρήσεις θα συγκεντρωθούν.

4) **Διαμόρφωση του περιβάλλοντος ελέγχου**

Προετοιμάζει το περιβάλλον δοκιμών πριν από την εκτέλεση. Επίσης, οργανώνει εργαλεία και άλλους πόρους

5) **Υλοποίηση σχεδιασμού ελέγχου**

Δημιουργήστε τις δοκιμές απόδοσης σύμφωνα με το σχεδιασμό δοκιμών σας.

6) **Εκτέλεση ελέγχου**

Εκτέλεση και παρακολούθηση των δοκιμών

7) **Ανάλυση και επανάληψη ελέγχου**

Τα αποτελέσματα που προκύπτουν αναλύονται και αν χρειαστεί επαναλαμβάνονται όσοι έλεγχοι επιθυμούμε έως ότου τα επίπεδα επίδοσης είναι τα μέγιστα δυνατά.

ΣΤ) Ο έλεγχος επίδοσης στο Web Testing

Ο έλεγχος επίδοσης εξασφαλίζει ότι η ιστοσελίδα μπορεί να δουλεύει ομαλά κάτω από οποιοδήποτε φόρτο. Περιλαμβάνει διαδικασίες όπως:

- Χρόνος απόκρισης της ιστοσελίδας σε πολλές και διαφορετικές ταχύτητες σύνδεσης(connection speeds)

- Έλεγχος φόρτου ώστε να γίνει αντιληπτό το πως συμπεριφέρεται η ιστοσελίδα κάτω από νορμάλ και αυξημένο φόρτο.
- Stress test στην ιστοσελίδα ώστε να εντοπιστεί το οριακό σημείο(break point) όταν ωθείται στα άκρα ο φόρτος που δέχεται.
- Έλεγχος για το πως ανακάμπτει η ιστοσελίδα έπειτα από κρασάρισμα λόγω του αυξημένου φόρτου.
- Έλεγχος για το αν οι τεχνικές βελτιστοποίησης λειτουργούν κανονικά ώστε να μειωθεί ο φόρτος χρόνου.

Ζ) Παράγοντες που εξετάζονται κατά τον έλεγχο επίδοσης

Ορισμένοι βασικοί παράγοντες είναι οι εξής:

- Χρήση επεξεργαστή
- Χρήση Μνήμης
- Χρόνος εκτέλεσης ανάγνωσης/εγγραφής στον δίσκο
- Εύρος Ζώνης(Bandwidth)
- Χρόνος Απόκρισης(Response time)
- Throughput
- Μέγιστος Αριθμός ενεργών συνεδριάσεων

Η) Εργαλεία Ελέγχου Επίδοσης

Υπάρχει μια μεγάλη ποικιλία εργαλείων δοκιμής απόδοσης που διατίθενται στην αγορά. Το εργαλείο που θα επιλέξετε για δοκιμή θα εξαρτηθεί από πολλούς παράγοντες όπως τύπους πρωτοκόλλου που υποστηρίζονται, κόστος άδειας, απαιτήσεις υλικού, υποστήριξη πλατφόρμας κ.λπ. Ακολουθεί μια λίστα με δημοφιλή εργαλεία δοκιμών.

• **LoadNinja*:*** Το LoadNinja φέρνει επανάσταση στον τρόπο με τον οποίο φορτώνουμε το τεστ. Αυτό το εργαλείο δοκιμής φόρτωσης που βασίζεται σε σύννεφο δίνει τη δυνατότητα στις ομάδες να καταγράφουν και να αναπαράγουν άμεσα ολοκληρωμένες δοκιμές φόρτωσης, χωρίς περίπλοκη δυναμική συσχέτιση και να

εκτελούν αυτές τις δοκιμές φόρτωσης σε πραγματικά προγράμματα περιήγησης σε κλίμακα. Οι ομάδες μπορούν να αυξήσουν την κάλυψη των δοκιμών. & περικοπή του χρόνου δοκιμής φορτίου πάνω από 60%

• *NeoLoad* :είναι η πλατφόρμα δοκιμών απόδοσης που έχει σχεδιαστεί για DevOps και ενσωματώνεται απρόσκοπτα στον υπάρχοντα αγωγό συνεχούς παράδοσης. Με το NeoLoad, οι ομάδες δοκιμάζουν 10 φορές γρηγορότερα από ό, τι με τα παραδοσιακά εργαλεία για να ικανοποιήσουν το νέο επίπεδο απαιτήσεων σε ολόκληρο τον κύκλο ζωής ανάπτυξης λογισμικού Agile - από συνιστώσες έως πλήρεις δοκιμές φόρτωσης σε όλο το σύστημα.

• *HP LoadRunner* :είναι ένα από τα πιο δημοφιλή εργαλεία δοκιμής απόδοσης στην αγορά σήμερα.Το Loadrunner διαθέτει μια εικονική γεννήτρια χρηστών που προσομοιώνει τις ενέργειες των ζωντανών ανθρώπινων χρηστών.

• *Jmeter*: ένα από τα κορυφαία εργαλεία που χρησιμοποιούνται για τη δοκιμή φόρτωσης διακομιστών ιστού και εφαρμογών.

I) Εν κατακλείδι

Στη Μηχανική Λογισμικού(Software Engineering), απαιτείται έλεγχος απόδοσης πριν από την εμπορία οποιουδήποτε προϊόντος λογισμικού. Διασφαλίζει την ικανοποίηση των πελατών και προστατεύει την επένδυση ενός επενδυτή από την αποτυχία προϊόντος. Το κόστος των δοκιμών απόδοσης είναι συνήθως περισσότερο από ό, τι αντιστοιχεί με βελτιωμένη ικανοποίηση, πίστη και διατήρηση των πελατών.

8) Usability Testing

A) Ορισμός

Ο **Έλεγχος Χρηστικότητας(Usability Test)** υπολογίζει το πόσο εύκολο είναι στην χρήση και φιλικό προς τον χρήστη(user-friendly) είναι ένα σύστημα λογισμικού. Αυτός ο έλεγχος εστιάζει κυρίως στην ευκολία του χρήστη να χρησιμοποιήσει την εφαρμογή, την ελαστικότητα όσον αφορά τους ελέγχους χειρισμού και στην ικανότητα του συστήματος να εκπληρώνει τους σκοπούς του. Ονομάζεται και Έλεγχος Εμπειρίας Χρήστη(User Experience Testing).

Β) Γιατί κάνουμε Έλεγχο Χρηστικότητας

Η αισθητική και ο σχεδιασμός είναι σημαντικά. Το πόσο καλά δείχνει οπτικά ένα προϊόν συνήθως υποδηλώνει και το πόσο καλά δουλεύει.

Υπάρχουν πολλές εφαρμογές/ιστοσελίδες λογισμικού, οι οποίες αποτυγχάνουν παταγωδώς μόλις γίνονται διαθέσιμες για κάποιους από τους παρακάτω λόγους:

-Που θα κάνω κλικ στην συνέχεια?

-Ποια εικόνα αντιπροσωπεύει τι?

-Μηνύματα σφάλματος δεν εμφανίζονται όταν πρέπει

-Ο χρόνος συνεδρίασης δεν είναι αρκετός

Έτσι ,ο έλεγχος χρηστικότητας εντοπίζει από νωρίς αυτά τα προβλήματα χρηστικότητας και μπορεί να σώσει ένα προϊόν από την αποτυχία.

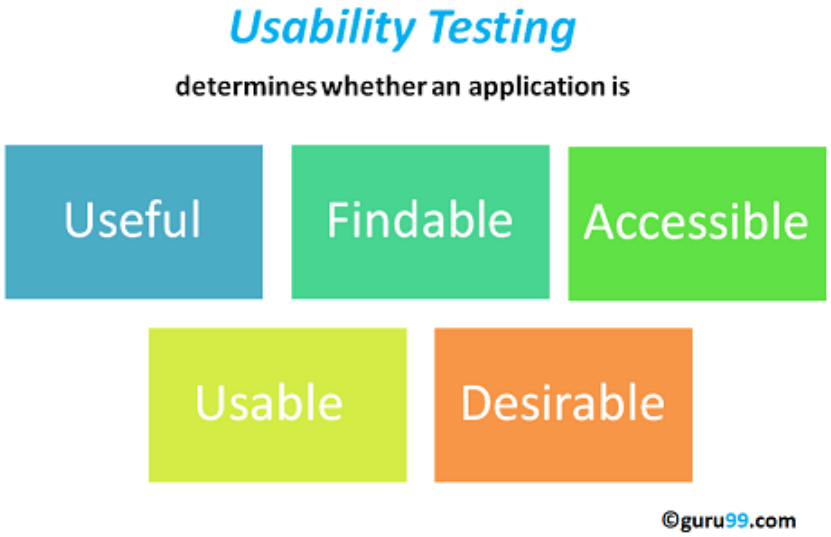
Γ) Παραδείγματα ελέγχου χρηστικότητας

- **Αποτελεσματικότητα του συστήματος**: Είναι εύκολη η εκμάθηση του συστήματος? Είναι το σύστημα χρήσιμο για το κοινό για το οποίο στοχεύει? Είναι το περιεχόμενο, τα χρώματα και τα εικονίδια κομψά?

- **Αποδοτικότητα**: Η πλοήγηση δεν πρέπει να είναι μεγάλη για να φτάσει ο χρήστης στην επιθυμητή σελίδα. Πρέπει, επιπλέον, να υπάρχει ομοιομορφία στο format των σελίδων της εφαρμογής/ιστοσελίδας .Τέλος πρέπει να υπάρχει επιλογή για αναζήτηση εντός της εφαρμογής/ιστοσελίδας

- **Ακρίβεια**: Δεν πρέπει να εμφανίζονται παρωχημένες ή λανθασμένες πληροφορίες ούτε σύνδεσμοι που να μην λειτουργούν(broken links).

- **Φιλικότητα προς τον χρήστη**: Οι χειρισμοί πρέπει είναι εύκολοι και εύχρηστοι, ενώ πρέπει να παρέχεται βοήθεια στους χρήστες για να καταλαβαίνουν τον τρόπο λειτουργία της ιστοσελίδας/εφαρμογής



Εικόνα 4: Τα χαρακτηριστικά που καθορίζουν μία εύχρηστη εφαρμογή

Δ) Πως γίνεται ο Έλεγχος Χρηστικότητας

Τα βήματα για να γίνει ένα πλήρης έλεγχος είναι τα εξής:

- **Σχεδιασμός** : Κατά την διάρκεια αυτού του σταδίου πρέπει να καθοριστούν οι στόχοι του ελέγχου που πρόκειται να γίνει. Πρέπει να καθοριστούν οι κρίσιμες λειτουργίες και οι στόχοι του συστήματος. Πρέπει, επίσης, να ανατεθούν επακριβώς οι αρμοδιότητες και το τι πρέπει να κάνει ο κάθε tester.
- **Πρόσληψη**: Στο στάδιο αυτό γίνεται η πρόσληψη των επιθυμητών tester όπως προβλέπεται από το σχέδιο του προηγούμενου βήματος.
- **Έλεγχος Χρηστικότητας** : Εκτελείται ο έλεγχος.
- **Ανάλυση Δεδομένων**: Τα δεδομένα, που συλλέγονται από τους ελέγχους, αναλύονται έτσι ώστε να προκύψουν χρήσιμα συμπεράσματα και να δοθούν προτάσεις για την βελτίωση της συνολικής χρηστικότητας του προιόντος.
- **Αναφορά**: Όλοι οι ενδιαφερόμενοι(designer,developer,CEO) ενημερώνονται για τις πληροφορίες και τα συμπεράσματα που προέκυψαν από τους ελέγχους.

Ε) Τρόποι διεξαγωγής

Δύο είναι οι κύριοι τρόποι διεξαγωγής είναι:

1) **Εργαστηριακός Έλεγχος Χρηστικότητας(Laboratory Usability Testing)**: Ο έλεγχος πραγματοποιείται σε ξεχωριστό μέρος παρουσία των παρατηρητών. Οι tester εκτελούν διάφορα τεστ. Ο ρόλος του παρατηρητή είναι να παρατηρήσει και να καταγράψει την συμπεριφορά και τις αντιδράσεις του tester.

2) **Απομακρυσμένος Έλεγχος Χρηστικότητας(Remote Usability Testing):** Στον έλεγχο αυτό οι παρατηρητές και οι testers δεν βρίσκονται στον ίδιο χώρο. Ένα αυτοματοποιημένο λογισμικό καταγράφει την συμπεριφορά του tester όπως την φωνή του, την δραστηριότητα και τις εκφράσεις του προσώπου του. Οι παρατηρητές αναλύουν αυτά τα δεδομένα.

ΣΤ) Τι πρέπει να ελεγχθεί

Για έναν αποτελεσματικό και επιτυχημένο έλεγχο πρέπει να πληρούνται ορισμένες προϋποθέσεις όπως:

- ✓ Ο έλεγχος πρέπει να ξεκινήσει στα αρχικά στάδια του σχεδιασμού και της ανάπτυξης του προιόντος
- ✓ Είναι καλή πρακτική να γίνει έλεγχος χρηστικότητας στο προϊόν του ανταγωνιστή προτού ξεκινήσει η ανάπτυξη του προιόντος. Αυτό θα βοηθήσει να γίνουν αντιληπτές οι προτιμήσεις του κοινού στο οποίο απευθύνεται το προϊόν.
- ✓ Πρέπει να γίνει η κατάλληλη επιλογή χρηστών για να τεστάρουν το σύστημα
- ✓ Πρέπει να μπουν περιορισμοί στο εύρος ζώνης ώστε να δοκιμαστεί κάτω από οποιαδήποτε ποιότητα δικτύου.
- ✓ Οι tester πρέπει να δώσουν βαρύτητα στις κρίσιμες λειτουργίες του συστήματος καθώς και σε αυτές που έχουν την μεγαλύτερη χρήση.
- ✓ Κάθε παρατηρητής πρέπει να παρατηρεί έναν και μόνο tester ώστε να γίνει ακριβέστερη καταγραφή της συμπεριφοράς του.

Ζ) Πλεονεκτήματα/ Μειονεκτήματα

Πλεονεκτήματα	**Μειονεκτήματα**
Βοηθάει στην αποκάλυψη προβλημάτων χρηστικότητας προτού αρχίσει το προϊόν να προωθείται στην αγορά.	Το κόστος αποτελεί σημαντικό ζήτημα στον έλεγχο χρηστικότητα. Χρειάζονται πολλοί πόροι για να στηθεί ένα εργαστήριο και επίσης το κόστος πρόσληψης tester είναι υψηλό.
Συμβάλει στην βελτίωση της ικανοποίησης του τελικού χρήστη	
Το σύστημα αποκτά υψηλή αποδοτικότητα και αποτελεσματικότητα	

Η) Ο Έλεγχος Χρηστικότητας στο Web Testing

Ο έλεγχος χρηστικότητας αποτελεί πλέον σημαντικό κομμάτι σε κάθε πρότζεκτ που σχετίζεται με τον ιστό. Μπορεί να γίνει από τυχαίους tester ή μια ομάδα ανθρώπων που αποτελούν το κοινό που στοχεύει αυτή η εφαρμογή.

Δύο τομείς που ελέγχονται είναι τα εξής:

1) **Πλοήγηση**: Το μενού, τα πλήκτρα ή οι σύνδεσμοι προς άλλες σελίδες πρέπει να είναι ευδιάκριτα σε όλες τις ιστοσελίδες.

2) **Περιεχόμενο**: Το περιεχόμενο πρέπει, επίσης , να είναι ευδιάκριτο χωρίς ορθογραφικά ή συντακτικά λάθη.

Εργαλεία που χρησιμοποιούνται: Chalkmark, Clicktale, Clixpy και Feedback Army

9) Database Testing

A) Ορισμός

Το database testing αποτελείται από μια πολυεπίπεδη διαδικασία, συμπεριλαμβανομένου του επιπέδου διεπαφής χρήστη (UI), του επιχειρησιακού επιπέδου, του επιπέδου πρόσβασης δεδομένων και της ίδιας της βάσης δεδομένων. Το επίπεδο UI ασχολείται με το σχεδιασμό διεπαφής της βάσης δεδομένων, ενώ το επίπεδο επιχειρήσεων περιλαμβάνει βάσεις δεδομένων που υποστηρίζουν επιχειρηματικές στρατηγικές.

Σαν ένα απλό ορισμό θα μπορούσαμε να πούμε ότι: Το database testing είναι ένας τύπος δοκιμών λογισμικού που ελέγχει το σχήμα, τους πίνακες, τους κανόνες ενεργοποίησης της υπό δοκιμή βάσης δεδομένων. Περιλαμβάνει τη δημιουργία σύνθετων ερωτημάτων για την εκτέλεση της δοκιμής φόρτωσης ή πίεσης στη βάση δεδομένων και τον έλεγχο της απόκρισης της. Ελέγχει την ακεραιότητα και τη συνέπεια των δεδομένων

B) Η αξία του Database Testing

Βάση Δεδομένων,η συλλογή διασυνδεδεμένων αρχείων σε διακομιστή, η αποθήκευση πληροφοριών, ενδέχεται να μην ασχολείται με τον ίδιο τύπο δεδομένων, δηλαδή οι βάσεις δεδομένων μπορεί να είναι ετερογενείς. Ως αποτέλεσμα, ενδέχεται να προκύψουν πολλά είδη σφαλμάτων εφαρμογής και ενοποίησης σε μεγάλα συστήματα βάσεων δεδομένων, τα οποία επηρεάζουν αρνητικά την απόδοση, την αξιοπιστία, τη συνέπεια και την ασφάλεια του συστήματος. Έτσι, είναι σημαντικό να δοκιμάσετε για να αποκτήσετε ένα σύστημα βάσης δεδομένων που ικανοποιεί τις ιδιότητες ACID (Ατομικότητα, Συνέπεια, Απομόνωση και Ανθεκτικότητα) ενός συστήματος διαχείρισης βάσης δεδομένων.

Ένα από τα πιο κρίσιμα επίπεδα είναι το επίπεδο πρόσβασης δεδομένων, το οποίο ασχολείται με βάσεις δεδομένων απευθείας κατά τη διάρκεια της διαδικασίας επικοινωνίας. Ο έλεγχος βάσης δεδομένων πραγματοποιείται κυρίως σε αυτό το επίπεδο και περιλαμβάνει στρατηγικές δοκιμών όπως ποιοτικό έλεγχο και διασφάλιση ποιότητας των βάσεων δεδομένων προϊόντων. Ο έλεγχος σε αυτά τα διαφορετικά επίπεδα χρησιμοποιείται συχνά για τη διατήρηση της συνοχής των συστημάτων βάσεων δεδομένων, τα οποία συνήθως εμφανίζονται στα ακόλουθα παραδείγματα:

Τα δεδομένα είναι κρίσιμα από επιχειρηματική άποψη. Εταιρείες όπως η Google ή η Symantec, οι οποίες συνδέονται με την αποθήκευση δεδομένων, πρέπει να διαθέτουν ένα σταθερό και σταθερό σύστημα βάσεων δεδομένων. Εάν εκτελούνται λειτουργίες βάσης δεδομένων όπως εισαγωγή, διαγραφή και ενημέρωση χωρίς να δοκιμαστεί πρώτα η βάση δεδομένων για συνέπεια, η εταιρεία κινδυνεύει να καταρρεύσει ολόκληρο το σύστημα

Ορισμένες εταιρείες έχουν διαφορετικούς τύπους βάσεων δεδομένων, καθώς και διαφορετικούς στόχους και αποστολές. Προκειμένου να επιτύχουν ένα επίπεδο λειτουργικότητας για την επίτευξη των εν λόγω στόχων, πρέπει να ελέγξουν το σύστημα βάσης δεδομένων τους

Η τρέχουσα προσέγγιση των δοκιμών ενδέχεται να μην είναι επαρκής στην οποία οι προγραμματιστές δοκιμάζουν επίσημα τις βάσεις δεδομένων. Ωστόσο, αυτή η προσέγγιση δεν είναι επαρκώς αποτελεσματική δεδομένου ότι οι προγραμματιστές βάσεων δεδομένων είναι πιθανό να επιβραδύνουν τη διαδικασία δοκιμών λόγω ελλείψεων επικοινωνίας. Μια ξεχωριστή ομάδα δοκιμών βάσης δεδομένων φαίνεται σκόπιμη.

Η δοκιμή βάσεων δεδομένων ασχολείται κυρίως με την εύρεση σφαλμάτων στις βάσεις δεδομένων έτσι ώστε να τα εξαλείψουν. Αυτό θα βελτιώσει την ποιότητα της βάσης δεδομένων ή του διαδικτυακού συστήματος.

Ο έλεγχος βάσης δεδομένων πρέπει να διακρίνεται από στρατηγικές για την αντιμετώπιση άλλων προβλημάτων, όπως σφάλματα της βάσης δεδομένων, σπασμένες εισαγωγές, διαγραφές ή ενημερώσεις. Εδώ, η αναδιαμόρφωση βάσεων δεδομένων είναι μια εξελικτική τεχνική που μπορεί να εφαρμοστεί

Γ) Τα βήματα του Database Testing

Το database testing ακολουθεί τα εξής βήματα:

•Data validity testing.

•Data Integrity testing

•Performance related to data base.

•Testing of Procedure, triggers and functions.

Ωστόσο τώρα προκύπτει το ερώτημα του τι πρέπει να ελέγξω σε μία βάση δεδομένων:

Μπορούμε να ελέγξουμε όλες τις λειτουργίες που συμβαίνουν σε κάθε ενέργεια που εκτελείται στην εφαρμογή. Οι ενέργειες μπορούν να περιλαμβάνουν διάφορες επιλογές όπως διαγραφή, προσθήκη ή αποθήκευση. Ελέγχει αν η προστιθέμενη εγγραφή προστίθεται στο database με την ακριβή τιμή. Ελέγχει ότι η διαγραμμένη εγγραφή αφαιρείται από τη βάση δεδομένων. Αυτοί είναι σημαντικοί ρόλοι που πρέπει να παρακολουθούνται σοβαρά.

Σήμερα η βάση δεδομένων γίνεται πιο περίπλοκη λόγω της επιχειρηματικής λογικής που παίζει σημαντικό ρόλο για τις εφαρμογές. Ο ελεγκτής πρέπει να βεβαιωθεί ότι οι τιμές έχουν προστεθεί σωστά μετά την εφαρμογή των επιχειρηματικών κανόνων.

Επομένως, το database testing είναι πραγματικά μια πολύπλοκη εργασία και θα πρέπει πάντα να εκτελείται μόνο εάν ο δοκιμαστής είναι πολύ έμπειρος σε αυτόν τον τομέα.

Δ) Τρόποι

Το database testing περιλαμβάνει 2 τρόπους τεσταρίσματος:

1)**Black Box Testing:** Που περιλαμβάνει

•data mapping

•Επαλήθευση αποθηκευμένων και ανακτημένων δεδομένων

•Χρησιμοποιεί τεχνικές δοκιμής μαύρου κουτιού όπως κατανομή ισοδυναμίας και ανάλυση οριακής τιμής

2)**White Box Testing**: Που περιλαμβάνει

•Testing database triggers and logical views

•Επικύρωση μοντέλων δεδομένων και σχήματος βάσης δεδομένων

•Έλεγχος ακεραιότητας αναφοράς και συνέπειας βάσης δεδομένων

•Χρησιμοποιεί τεχνικές δοκιμής white box όπως κάλυψη δηλώσεων, κάλυψη αποφάσεων, κάλυψη συνθηκών

Ε) Εργαλεία

1) **SolarWinds Database Performance Analyzer**

• Είναι ένα λογισμικό διαχείρισης βάσεων δεδομένων που μπορεί να εκτελέσει παρακολούθηση, ανάλυση και συντονισμό ερωτημάτων SQL.

• Κάνει τη χρήση μηχανικής μάθησης για ανίχνευση ανωμαλιών.

• Παρέχει υποστήριξη σε βάσεις δεδομένων πολλαπλών πλατφορμών για cloud καθώς και για το εσωτερικό περιβάλλον.

• Διαθέτει έναν σύμβουλο ευρετηρίου και συντονισμού ερωτημάτων για να παρέχει συμβουλές από ειδικούς.

2) **Data Factory**

Είναι ένα εμπορικό εργαλείο δοκιμών βάσεων δεδομένων που λειτουργεί ως γεννήτρια δεδομένων και διαχειριστής δεδομένων για τη δοκιμή βάσεων δεδομένων.

• Αυτό το εργαλείο εξυπηρετήθηκε με καινοτόμο και εύχρηστο περιβάλλον εργασίας χρήστη και ικανό να διαχειριστεί πολύπλοκες σχέσεις δεδομένων.

• Αυτό είναι πιο αποτελεσματικό για τον χειρισμό ερωτημάτων με μεγάλο αριθμό δεδομένων.

• Παρέχει ευκολία εκτέλεσης δοκιμών πίεσης ή φόρτωσης στη βάση δεδομένων.

3)**SQL Server**

Χρησιμοποιούνται εργαλεία για την πραγματοποίηση δοκιμών μονάδας

• Το εμπορικό εργαλείο στο οποίο δημιουργούνται δοκιμές σε έργα VB ή C #.

• Οι δοκιμές δημιουργούνται με δύο τρόπους, όπως είτε δημιουργώντας δοκιμές από τη βάση δεδομένων χρησιμοποιώντας σενάριο T-SQL ή μπορείτε να προσθέσετε δοκιμές με μη αυτόματο τρόπο χρησιμοποιώντας πρότυπα.

• Κατά τη δημιουργία δοκιμών από ένα έργο βάσης δεδομένων, μπορείτε να χρησιμοποιήσετε τον SQL Server Object Explorer

4)**Oracle SQL Developer**

Λειτουργεί με παρόμοιο τρόπο με αυτόν του SQL Developer.

• Είναι ένα εργαλείο δοκιμής βάσης δεδομένων που χρησιμοποιείται για τη βάση δεδομένων του Oracle Cloud.

• Τα στοιχεία της Oracle περιλαμβάνουν το Oracle Web Agent και συνεργάζονται με IBM DB2, Microsoft Access, MySQL, Sybase και Teradata.

ΣΤ) To Database testing στο Web Testing

Η βάση δεδομένων αποτελεί αναπόσπαστο μέρος μιας εφαρμογής ιστού και πρέπει να ελέγχεται διεξοδικά. Οι έλεγχοι αυτοί περιλαμβάνουν:

- Έλεγχος για εμφάνιση σφαλμάτων κατά την εκτέλεση επερωτήσεων
- Η ακεραιότητα των δεδομένων διατηρείται κατά την δημιουργία, ενημέρωση και διαγραφή δεδομένων σε μία βάση δεδομένων.
- Έλεγχος για χρόνο απόκρισης των επερωτήσεων.
- Έλεγχος αν τα δεδομένα, που ανακτήθηκαν από την βάση δεδομένων, εμφανίζονται κανονικά στην εφαρμογή

10) Crowd Testing

A) Ορισμός

Το Crowd Testing(ή Crowdsource Testing) σημαίνει ότι δοκιμάζουμε και βελτιστοποιούμε τη φιλικότητα προς το χρήστη, τη χρηστικότητα και τη λειτουργικότητα των ψηφιακών προϊόντων σας υπό πραγματικές συνθήκες χρησιμοποιώντας τη συλλογική γνώση μιας παγκόσμιας διαδικτυακής κοινότητας - του πλήθους μας.

B) To Crowd Testing στο Web Testing

Επιλέγεται ένας τεράστιος αριθμός ατόμων(πλήθος) προκειμένου να εκτελέσουν ελέγχους που σε διαφορετική περίπτωση θα έπρεπε να εκτελεστούν από μία ομάδα ανθρώπων της εταιρείας. Το crowd testing είναι ένα ενδιαφέρον concept και βοηθάει στον εντοπισμό πολλών απαρατήρητων αποκλίσεων.

Γ) Πλεονεκτήματα/ Μειονεκτήματα

Πλεονεκτήματα	**Μειονεκτήματα**
Η βασική ομάδα core testing ενδέχεται να μην έχει όλους τους πόρους για τη δοκιμή του λογισμικού σε διαφορετικά περιβάλλοντα και σε διαφορετικές καταστάσεις (π.χ. διαφορετικά εύρη ζώνης Διαδικτύου, συσκευές κ.λπ.), καθώς ενδέχεται να μην είναι δυνατόν να υπάρχουν όλοι οι πόροι για τη δημιουργία διαφορετικών περιβαλλόντων στα οποία το λογισμικό πρέπει να δοκιμαστεί.	Η εμπιστευτικότητα πρέπει να αντιμετωπίζεται στενά καθώς αυξάνεται ο αριθμός των μη εσωτερικών ατόμων που εξετάζουν το υπό δοκιμή σύστημα.
Είναι οικονομικά αποδοτικό, καθώς η εταιρεία προϊόντων πληρώνει μόνο για τα έγκυρα σφάλματα που αναφέρονται. Συνήθως ο χρόνος δοκιμής του λογισμικού είναι συγκριτικά μικρότερος, επομένως οδηγεί σε καλύτερη παραγωγικότητα και επομένως είναι φθηνότερος από την πρόσληψη μηχανικών, σχεδιαστών και ειδικών.	Οι υπεύθυνοι δοκιμών Crowdsource που αποζημιώνονται βάσει του αριθμού των σφαλμάτων που εντοπίστηκαν ενδέχεται να εντοπίσουν μεγαλύτερο αριθμό σφαλμάτων με μικρότερο αντίκτυπο ενώ παρακάμπτοντας τα πιο κρίσιμα ή πιο δύσκολα να αντιγράψουν σφάλματα.
Η ομάδα των υπευθύνων δοκιμών είναι διαφορετική με παραλλαγές σε γλώσσες καθώς και σε τοπικές ρυθμίσεις. Αυτό βοηθά στη δοκιμή εφαρμογών που βασίζονται στον εντοπισμό.	Οι δοκιμές Crowdsource θα οδηγήσουν σε αυξημένη ανάγκη για εποπτεία διαχείρισης λόγω των διαφορών στις ζώνες ώρας και τις τοποθεσίες των δοκιμαστών, τις γλώσσες και τους πολιτισμούς.
Καθώς υπάρχει μεγάλος αριθμός δοκιμαστών που δοκιμάζουν ένα λογισμικό ταυτόχρονα, οι δοκιμές μπορούν να γίνουν γρήγορα, με αποτέλεσμα λιγότερο χρόνο στην αγορά.	Η διασφάλιση της κάλυψης των δοκιμών στις δοκιμές crowdsource μπορεί να είναι δύσκολη καθώς οι δοκιμές δεν προγραμματίζονται ή παρακολουθούνται με τον ίδιο τρόπο όπως οι παραδοσιακές προσπάθειες καταρράκτη ή δοκιμής Agile.

Δ) Φάσεις

Το Crowdtesting ακολουθεί μερικές διαδοχικές φάσεις. Σε αντίθεση με τις εσωτερικές δοκιμές, ορισμένες επιπλέον λεπτομέρειες πρέπει να ληφθούν υπόψη καθώς εμπλέκονται εξωτερικοί ελεγκτές. Εδώ είναι οι τέσσερις φάσεις της εφαρμογής crowdtesting.

Φάση Ι - Φάση προγραμματισμού και προετοιμασίας

Προσδιορίστε το εύρος δοκιμών: Το Crowdtesting μπορεί να γίνει σε ένα ανεπτυγμένο προϊόν, δυνατότητα, μερική ενότητα ή πρωτότυπο. Μπορείτε να καθορίσετε περιοχές "σε πεδίο εφαρμογής" και "εκτός πεδίου".

Παροχή του φόντου: Το Crowdtesting περιλαμβάνει εξωτερικούς δοκιμαστές τις περισσότερες φορές. Πρέπει να διατηρήσετε έτοιμες τις επιχειρηματικές απαιτήσεις και τους στόχους δοκιμής. Το Crowdtesters θα είναι σε θέση να κατανοήσει τις δοκιμαστικές δραστηριότητες και να αποδώσει καλύτερα.

Προσδιορισμός σετ δεξιοτήτων δοκιμαστών: Πρέπει να καθορίσετε το σετ δεξιοτήτων που απαιτείται για το crowdtesters. Παράδειγμα - εάν πρέπει να δοκιμάσετε μια τραπεζική αίτηση, φυσικά θα περιμένετε έναν δοκιμαστή με γνώσεις τραπεζικής διαδικασίας. Ο καθορισμός δεξιοτήτων δοκιμών βοηθά στην επιλογή των σωστών πλήθους.

Δημιουργία δοκιμαστικών σχεδίων: Τα Crowdtests θα πρέπει να προγραμματιστούν σύμφωνα με τα σχέδια. Μπορεί να σχεδιάζετε την κυκλοφορία ενός προϊόντος πριν από τις χριστουγεννιάτικες αγορές. Το σχέδιό σας θα περιέχει λεπτομέρειες προγράμματος και τοποθεσίας.

Καθορίστε προϋπολογισμούς και παραδοτέα: Πρέπει να προετοιμάσετε τον προϋπολογισμό για crowdtesting. Προετοιμάστε την ομάδα crowdtesting. Βεβαιωθείτε ότι οι προσδοκίες και τα παραδοτέα είναι σαφώς καθορισμένα. Καθορίστε παραδοτέα όπως δοκιμαστικές αναφορές και άλλες μετρήσεις. Μπορείτε να καθορίσετε ένα σύστημα αναφοράς σφαλμάτων.

Σημείο επαφής: Δημιουργία συντονιστή για την επίβλεψη της δραστηριότητας crowdtesting. Αυτό μπορεί να είναι υπεύθυνος δοκιμών από τον οργανισμό σας. Παράλληλα, μπορείτε να έχετε έναν συντονιστή από την εταιρεία crowdtesting.

Φάση ΙΙ - Φάση έναρξης και διαμόρφωσης

Δοκιμές και διαμορφώσεις: Σε αυτήν τη φάση, θα προετοιμάσετε όλες τις δοκιμαστικές σας περιπτώσεις. Διαμορφώστε λειτουργικά σενάρια και απαιτούμενα δεδομένα για την εκτέλεση δοκιμαστικών περιπτώσεων. Πρέπει να γίνουν τεχνικές διαμορφώσεις σε περιβάλλον εφαρμογής και δοκιμής.

Πρόσβαση: Τα διαπιστευτήρια πρόσβασης για τα crowdtesters πρέπει να ρυθμιστούν με ασφάλεια. Το περιβάλλον δοκιμής πρέπει να διαμορφωθεί ώστε να παρακολουθεί τη δοκιμαστική δραστηριότητα.

Δοκιμαστικός γύρος πλήθους δοκιμών: Μπορείτε να ρυθμίσετε μια δοκιμαστική φάση δοκιμών με crowdtesters. Θα μετρήσει την ετοιμότητα για δοκιμή με μερικά αποτελέσματα δείγματος.

Φάση III - Φάση εκτέλεσης

Μόλις είστε έτοιμοι, μπορεί να ξεκινήσει η εκτέλεση του πλήθους δοκιμών. Θα πρέπει να επιβλέπετε την εκτέλεση του πλήθους.

Φάση IV - Αξιολόγηση αποτελεσμάτων

Μετά την εκτέλεση, τα αποτελέσματα των δοκιμών μπορούν να επικυρωθούν για την αξιολόγηση της απόδοσης. Τα σφάλματα μπορούν να ταξινομηθούν ανάλογα με τη σοβαρότητα και να αντιμετωπιστούν.

Ε) Πότε να μην χρησιμοποιείται το Crowd Testing

1)Πρωτοποριακές τεχνικές και στρατηγικές

Για να είναι ανταγωνιστικοί, οι οργανισμοί κυκλοφορούν νέα χαρακτηριστικά και καινοτομίες. Τα προϊόντα που έχουν αυτά δεν μπορούν να δοκιμαστούν. Υπάρχει η δυνατότητα του ανταγωνισμού να εκμεταλλευτεί τις στρατηγικές της εταιρείας. Η εταιρική κατασκοπεία μπορεί να γίνει ευκολότερη όταν το προϊόν είναι γεμάτο δοκιμές. Επομένως, είναι καλύτερο να δοκιμάσετε μια ομάδα QA στο εσωτερικό σας για να δοκιμάσετε αυτά τα καινοτόμα χαρακτηριστικά

2)Εμπιστευτικότητα δεδομένων

Οι εφαρμογές που περιέχουν ή έχουν πρόσβαση σε εμπιστευτικά δεδομένα με οποιονδήποτε τρόπο δεν είναι κατάλληλες για πλήθος δοκιμών. Οι κίνδυνοι για την ασφάλεια στον κυβερνοχώρο μπορεί να είναι υψηλοί σε αυτά τα σενάρια εάν είναι γεμάτο δοκιμές.

3) Fitment for crowdtesting

Όλες οι δοκιμαστικές περιπτώσεις δεν είναι καλοί υποψήφιοι για πλήθος δοκιμών. Ένα καλά καθορισμένο πρόγραμμα QA θα έχει μια στρατηγική πολλαπλών αξόνων για το QA. Θα περιλαμβάνει μη αυτόματες δοκιμές και αυτοματοποιημένες δοκιμές. Το Crowdtesting είναι μια άλλη μέθοδος για να διασφαλιστεί ότι όλες οι λειτουργίες έχουν δοκιμαστεί πριν από την κυκλοφορία. Ένας διευθυντής QA με εμπειρία μπορεί να εντοπίσει αυτές τις δοκιμαστικές περιπτώσεις.

ΣΤ) Πως να επιλέξουμε την σωστή πλατφόρμα

Οι πλατφόρμες δοκιμών πλήθους είναι πολλές. Θα πρέπει να επιλέξετε τη σωστή πλατφόρμα crowdtesting για καλύτερα αποτελέσματα.

Επιλέξτε με βάση την εμπειρία σε παρόμοια έργα. Πρέπει να επιλέξετε πλατφόρμες σύμφωνα με την απαίτησή σας.

Πρέπει να καθοριστεί η συμβατότητα της πλατφόρμας Crowdtest με διάφορα λογισμικά. Θα χρειαστεί να καταγράψετε αναφορές σφαλμάτων οπτικού περιεχομένου. Τα crowdtesters βρίσκονται σε διαφορετικές τοποθεσίες. Χρειάζεστε ένα ικανό εργαλείο που να αποτυπώνει αποτελεσματικά τα αποτελέσματα.

Z) Τι κοιτάμε σε μια πλατφόρμα

✔ **Εταιρείες με εμπειρία στο crowd testing**: Η εμπειρία μετράει ενώ κάνετε μια επιλογή σε πλατφόρμες crowdtesting. Το Crowdtesting έχει τις προκλήσεις του στη διαχείριση κατανεμημένων ομάδων και επιχειρήσεων. Πρέπει να εξασφαλίσετε αποτελέσματα υψηλής ποιότητας επιλέγοντας μια έμπειρη εταιρεία.

✔ **Η φύση του project**: Μπορείτε να αναζητήσετε εταιρείες που μπορούν να εξυπηρετήσουν τις ανάγκες σας. Μπορεί να απαιτήσετε τη διενέργεια δοκιμών σε πολλές χώρες. Επιλέξτε εταιρείες που έχουν εξειδίκευση στην παροχή παγκόσμιων έργων crowdtesting. Ορισμένες εταιρείες μπορεί να είναι πιο έμπειρες σε εφαρμογές για κινητά. Άλλες εταιρείες μπορεί να έχουν εμπειρία σε εταιρικές εφαρμογές. Πρέπει να επιλέξετε μια κατάλληλα έμπειρη εταιρεία crowdtesting

✔ **Σταθερές οικονομικά εταιρείες**: Συνιστάται να συνεργαστείτε με οικονομικά σταθερές εταιρείες. Θα φέρει εμπιστοσύνη στη διασταυρούμενη γεωγραφία και στις παγκόσμιες δυνατότητες εκτέλεσης δοκιμών.

References

Crowdsourced testing. (21 January 2020 p.). Retrieved from wikipedia.org: https://en.wikipedia.org/wiki/Crowdsourced_testing

Crowdtesting – all you need to know. (2018). Retrieved from usersnap.com: https://usersnap.com/blog/crowdtesting-all-you-need-to-know/

Database(Data) Testing Tutorial with Sample TestCases. (16 April 2020 p.). Retrieved from guru99.com: https://www.guru99.com/data-testing.html

Performance Testing Tutorial: What is, Types, Metrics & Example. (13 March 2020 p.). Retrieved from guru99.com: https://www.guru99.com/performance-testing.html

Web Application Testing Checklist: Example Test Cases for Website. (9 April 2020 p.). Retrieved from guru99.com: https://www.guru99.com/complete-web-application-testing-checklist.html

Web Application Testing: 8 Step Guide to Website Testing. (2020, April 15). Retrieved from guru99.com: https://www.guru99.com/web-application-testing.html

What is Compatibility Testing? Forward & Backward Testing (Example). (4 April 2020 p.). Retrieved from guru99.com: https://www.guru99.com/compatibility-testing.html

What is Functional Testing? Types & Examples (Complete Tutorial). (12 April 2020 p.). Retrieved from guru99.com: https://www.guru99.com/web-application-testing.html

What is Interface Testing? Types & Example. (4 April 2020 p.). Retrieved from guru99.com: https://www.guru99.com/interface-testing.html

What is Security Testing? Types with Example. (12 April 2020 p.). Retrieved from з guru99.com: https://www.guru99.com/what-is-security-testing.html

What is Usability Testing? UX(User Experience) Testing Example. (16 April 2020 p.). Отримано з guru99.com: https://www.guru99.com/usability-testing-tutorial.html