Testing Web Applications

Testing Quality Dimensions-I

- *Content* dievaluasi pada level sintaksis dan semantik..
  - syntactic level—ejaan, tanda baca, dan tata bahasa dinilai untuk dokumen berbasis teks.
  - semantic level—kebenaran (informasi yang disajikan), konsistensi (di seluruh objek konten dan objek terkait) dan kurangnya ambiguitas semuanya dinilai.
- *Function* diuji untuk kebenaran, ketidakstabilan, dan kesesuaian umum dengan standar implementasi yang sesuai (mis., Standar bahasa Java atau XML).
- *Structure* dinilai untuk memastikannya
  - memberikan konten dan fungsi WebApp dengan benar
  - dapat diperpanjang
  - dapat didukung saat konten atau fungsionalitas baru ditambahkan.

Testing Quality Dimensions-II

- *Usability* diuji untuk memastikan bahwa setiap kategori pengguna
  - didukung oleh antarmuka
  - dapat mempelajari dan menerapkan semua sintaks dan semantik navigasi yang diperlukan
- *Navigability* navigasi diuji untuk memastikan hal itu
  - semua sintaksis navigasi dan semantik dilakukan untuk mengungkap kesalahan navigasi (mis., tautan mati, tautan tidak patut, tautan salah).
- *Performance* diuji dalam berbagai kondisi operasi, konfigurasi, dan pemuatan untuk memastikan hal itu
  - sistem responsif terhadap interaksi pengguna
  - sistem menangani pemuatan ekstrem tanpa degradasi operasional yang tidak dapat diterima

Testing Quality Dimensions-III

- *Compatibility* diuji dengan mengeksekusi WebApp dalam berbagai konfigurasi host yang berbeda pada sisi klien dan server.
  - Tujuannya adalah untuk menemukan kesalahan yang spesifik untuk konfigurasi host yang unik.
- *Interoperability* diuji untuk memastikan bahwa WebApp berinteraksi dengan baik dengan aplikasi dan / atau basis data lainnya.
- *Security* diuji dengan menilai potensi kerentanan dan mencoba untuk mengeksploitasi masing-masing.
  - Setiap upaya penetrasi yang berhasil dianggap sebagai kegagalan keamanan.

Errors in a WebApp

- Karena banyak jenis tes WebApp mengungkap masalah yang pertama kali dibuktikan di sisi klien, Anda sering melihat gejala kesalahan, bukan kesalahan itu sendiri.
- Karena WebApp diimplementasikan dalam sejumlah konfigurasi yang berbeda dan dalam lingkungan yang berbeda, mungkin sulit atau tidak mungkin untuk mereproduksi kesalahan di luar lingkungan di mana kesalahan itu awalnya ditemui.
- Meskipun beberapa kesalahan adalah hasil dari desain yang salah atau HTML yang tidak benar (atau bahasa pemrograman lain) pengkodean, banyak kesalahan dapat ditelusuri ke konfigurasi WebApp.
- Karena WebApps berada di dalam arsitektur klien / server, kesalahan bisa sulit dilacak di tiga lapisan arsitektur: klien, server, atau jaringan itu sendiri.
- Beberapa kesalahan disebabkan oleh lingkungan operasi statis (mis., Konfigurasi khusus tempat pengujian dilakukan), sementara yang lain disebabkan oleh lingkungan operasi dinamis (mis., Pemuatan sumber daya sesaat atau kesalahan terkait waktu).

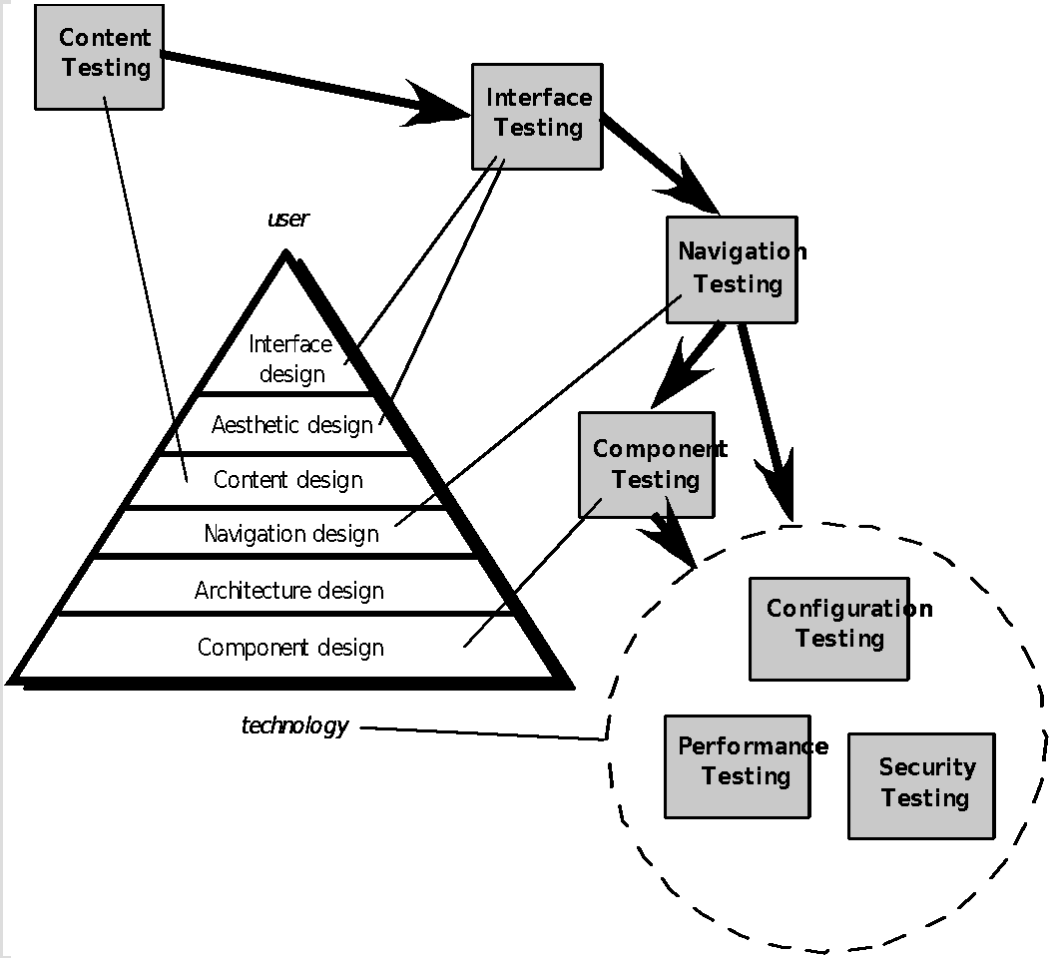
WebApp Testing Strategy-I

- Model konten untuk WebApp ditinjau untuk mengungkap kesalahan.
- Model antarmuka ditinjau untuk memastikan bahwa semua kasus penggunaan dapat ditampung.
- Model desain untuk WebApp ditinjau untuk mengungkap kesalahan navigasi.
- Antarmuka pengguna diuji untuk mengungkap kesalahan dalam presentasi dan / atau mekanisme navigasi.
- Komponen fungsional yang dipilih diuji unit.

WebApp Testing Strategy-II

- Navigasi di seluruh arsitektur diuji.
- WebApp diimplementasikan dalam berbagai konfigurasi lingkungan yang berbeda dan diuji kompatibilitasnya dengan setiap konfigurasi.
- Tes keamanan dilakukan dalam upaya untuk mengeksploitasi kerentanan di WebApp atau di dalam lingkungannya.
- Tes kinerja dilakukan.
- WebApp diuji oleh populasi pengguna akhir yang terkontrol dan dipantau
- hasil interaksi mereka dengan sistem dievaluasi untuk kesalahan konten dan navigasi, masalah kegunaan, masalah kompatibilitas, dan keandalan dan kinerja WebApp.

The Testing Process



Content Testing

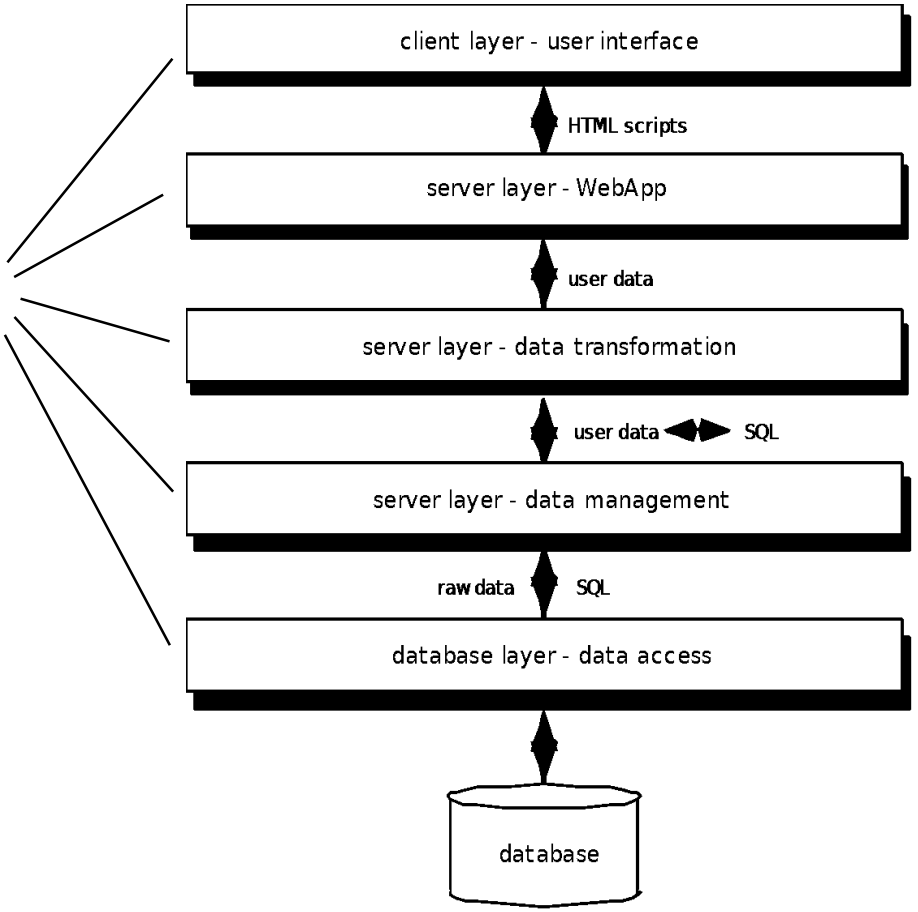
- Pengujian konten memiliki tiga tujuan penting:
  - untuk mengungkap kesalahan sintaksis (mis., kesalahan ketik, kesalahan tata bahasa) dalam dokumen berbasis teks, representasi grafis, dan media lainnya
  - untuk mengungkap kesalahan semantik (mis., kesalahan dalam keakuratan atau kelengkapan informasi) di setiap objek konten yang disajikan saat navigasi terjadi, dan
  - untuk menemukan kesalahan dalam organisasi atau struktur konten yang disajikan kepada pengguna akhir.

Assessing Content Semantics

- Apakah informasinya akurat secara faktual?
- Apakah informasinya singkat dan langsung ke sasaran?
- Apakah tata letak objek konten mudah dipahami pengguna?
- Dapatkah informasi yang disematkan dalam objek konten mudah ditemukan?
- Sudahkah referensi yang tepat diberikan untuk semua informasi yang berasal dari sumber lain?
- Apakah informasi yang disajikan konsisten secara internal dan konsisten dengan informasi yang disajikan dalam objek konten lain?
- Apakah kontennya menyinggung, menyesatkan, atau membuka pintu ke pengadilan?
- Apakah konten tersebut melanggar hak cipta atau merek dagang yang ada?
- Apakah konten mengandung tautan internal yang melengkapi konten yang ada? Apakah tautannya benar?
- Apakah gaya estetika konten bertentangan dengan gaya estetika antarmuka?

Database Testing

Tests are defined for each layer



User Interface Testing

- Fitur antarmuka diuji untuk memastikan bahwa aturan desain, estetika, dan konten visual terkait tersedia untuk pengguna tanpa kesalahan.
- Mekanisme antarmuka individual diuji dengan cara yang analog dengan pengujian unit.
- Setiap mekanisme antarmuka diuji dalam konteks use case atau NSU untuk kategori pengguna tertentu.
- Antarmuka lengkap diuji terhadap kasus penggunaan yang dipilih dan NSU untuk mengungkap kesalahan dalam semantik antarmuka.
- Antarmuka diuji dalam berbagai lingkungan (mis., Browser) untuk memastikan bahwa itu akan kompatibel.

NSU = Navigation Semantic Units

Testing Interface Mechanisms-I

- *Links*— mekanisme navigasi yang menautkan pengguna ke beberapa objek atau fungsi konten lainnya.
- *Forms*— dokumen terstruktur yang berisi bidang kosong yang diisi oleh pengguna. Data yang terkandung dalam bidang digunakan sebagai input ke satu atau beberapa fungsi WebApp.
- *Client-side scripting*— daftar perintah yang diprogram dalam bahasa skrip (mis., Javascript) yang menangani input informasi melalui formulir atau interaksi pengguna lainnya
- *Dynamic HTML*— mengarah ke objek konten yang dimanipulasi di sisi klien menggunakan scripting atau cascading style sheets (CSS).
- *Client-side pop-up windows*— jendela kecil yang muncul tanpa interaksi pengguna. Jendela ini dapat berorientasi konten dan mungkin memerlukan beberapa bentuk interaksi pengguna.

Testing Interface Mechanisms-II

- *CGI scripts*— skrip Common Gateway Interface (CGI) mengimplementasikan metode standar yang memungkinkan server Web untuk berinteraksi secara dinamis dengan pengguna (misalnya, WebApp yang berisi formulir dapat menggunakan skrip CGI untuk memproses data yang terkandung dalam formulir setelah diserahkan oleh pengguna).
- *Streaming content*— alih-alih menunggu permintaan dari sisi klien, objek konten diunduh secara otomatis dari sisi server. Pendekatan ini kadang-kadang disebut teknologi "push" karena server mendorong data ke klien.
- *Cookies*— blok data yang dikirim oleh server dan disimpan oleh browser sebagai konsekuensi dari interaksi pengguna tertentu. Konten data adalah khusus WebApp (mis., Data identifikasi pengguna atau daftar item yang telah dipilih untuk dibeli oleh pengguna).
- *Application specific interface mechanisms*— termasuk satu atau lebih mekanisme antarmuka "makro" seperti keranjang belanja, pemrosesan kartu kredit, atau kalkulator biaya pengiriman.

Usability Tests

- Design by WebE team … dijalankan oleh pengguna akhir
- Testing sequence …
  - Tentukan satu set kategori pengujian kegunaan dan identifikasi tujuan untuk masing-masing.
  - Tes desain yang akan memungkinkan setiap tujuan dievaluasi.
  - Pilih peserta yang akan melakukan tes.
  - Interaksi instrumen peserta dengan WebApp saat pengujian dilakukan.
  - Kembangkan mekanisme untuk menilai kegunaan WebApp
- Different levels of abstraction:
  - kegunaan mekanisme antarmuka spesifik (mis., formulir) dapat dinilai
  - kegunaan halaman Web lengkap (mencakup mekanisme antarmuka, objek data dan fungsi terkait) dapat dievaluasi
  - kegunaan dari WebApp yang lengkap dapat dipertimbangkan..

Compatibility Testing

- Pengujian kompatibilitas adalah untuk menetapkan serangkaian konfigurasi komputasi sisi klien dan variasinya
- Buat struktur pohon yang mengidentifikasi
  - setiap platform komputasi
  - perangkat tampilan khas
  - sistem operasi yang didukung pada platform
  - browser tersedia
  - kemungkinan kecepatan koneksi internet
  - informasi serupa.
- Turunkan serangkaian uji validasi kompatibilitas
  - berasal dari tes antarmuka yang ada, tes navigasi, tes kinerja, dan tes keamanan.
  - maksud dari tes ini adalah untuk mengungkap kesalahan atau masalah eksekusi yang dapat ditelusuri ke perbedaan konfigurasi.

Component-Level Testing

- Berfokus pada serangkaian tes yang mencoba mengungkap kesalahan dalam fungsi WebApp
- Metode perancangan kotak hitam dan kotak putih konvensional dapat digunakan
- Pengujian basis data sering merupakan bagian integral dari aturan pengujian komponen

Navigation Testing

- Mekanisme navigasi berikut harus diuji:
  - *Navigation links*— mekanisme ini mencakup tautan internal di dalam WebApp, tautan eksternal ke WebApps lain, dan jangkar di dalam halaman Web tertentu.
  - *Redirects*— tautan ini mulai digunakan ketika pengguna meminta URL yang tidak ada atau memilih tautan yang tujuannya telah dihapus atau yang namanya telah berubah.
  - *Bookmarks*— meskipun bookmark adalah fungsi browser, WebApp harus diuji untuk memastikan bahwa judul halaman yang bermakna dapat diekstraksi saat bookmark dibuat.
  - *Frames and framesets*— diuji untuk konten yang benar, tata letak dan ukuran yang tepat, kinerja unduhan, dan kompatibilitas browser
  - *Site maps*— Setiap entri peta situs harus diuji untuk memastikan bahwa tautan membawa pengguna ke konten atau fungsionalitas yang tepat.
  - *Internal search engines*— Pengujian mesin pencari memvalidasi keakuratan dan kelengkapan pencarian, sifat penanganan kesalahan mesin pencari, dan fitur pencarian lanjutan

Testing Navigation Semantics-I

- Apakah NSU tercapai secara keseluruhan tanpa kesalahan?
- Apakah setiap simpul navigasi (ditentukan untuk NSU) dapat dijangkau dalam konteks jalur navigasi yang ditentukan untuk NSU?
- Jika NSU dapat dicapai dengan menggunakan lebih dari satu jalur navigasi, apakah setiap jalur yang relevan telah diuji?
- Jika panduan disediakan oleh antarmuka pengguna untuk membantu dalam navigasi, apakah arahnya benar dan dapat dimengerti saat navigasi berlanjut?
- Apakah ada mekanisme (selain panah ‘back' browser) untuk kembali ke simpul navigasi sebelumnya dan ke awal jalur navigasi.
- Apakah mekanisme untuk navigasi di dalam simpul navigasi yang besar (mis., halaman web yang panjang) berfungsi dengan baik?
- Jika suatu fungsi akan dieksekusi pada sebuah node dan pengguna memilih untuk tidak memberikan input, dapatkah sisa NSU diselesaikan?

Testing Navigation Semantics-II

- Jika suatu fungsi dieksekusi pada suatu simpul dan suatu kesalahan dalam pemrosesan fungsi terjadi, dapatkah NSU diselesaikan?
- Apakah ada cara untuk menghentikan navigasi sebelum semua node tercapai, tetapi kemudian kembali ke tempat navigasi dihentikan dan dilanjutkan dari sana?
- Apakah setiap node dapat dijangkau dari peta situs? Apakah nama simpul bermakna bagi pengguna akhir?
- Jika sebuah simpul dalam NSU dicapai dari beberapa sumber eksternal, apakah mungkin untuk memproses ke simpul berikutnya di jalur navigasi. Apakah mungkin untuk kembali ke simpul sebelumnya di jalur navigasi?
- Apakah pengguna memahami lokasinya dalam arsitektur konten saat NSU dijalankan?

Configuration Testing

- Server-side
  - Apakah WebApp sepenuhnya kompatibel dengan OS server?
  - Apakah file sistem, direktori, dan data sistem terkait dibuat dengan benar ketika WebApp sedang operasional?
  - Apakah langkah-langkah keamanan sistem (mis., Firewall atau enkripsi) memungkinkan WebApp untuk mengeksekusi dan melayani pengguna tanpa gangguan atau penurunan kinerja?
  - Apakah WebApp telah diuji dengan konfigurasi server terdistribusi (jika ada) yang telah dipilih?
  - Apakah WebApp terintegrasi dengan perangkat lunak database? Apakah WebApp sensitif terhadap berbagai versi perangkat lunak database?
  - Apakah skrip WebApp sisi server dijalankan dengan benar?
  - Apakah kesalahan administrator sistem telah diperiksa untuk pengaruhnya pada operasi WebApp?
  - Jika server proxy digunakan, apakah perbedaan dalam konfigurasinya telah diatasi dengan pengujian di tempat?

Configuration Testing

- Client-side
  - *Hardware*—CPU, memory, storage and printing devices
  - *Operating systems*—Linux, Macintosh OS, Microsoft Windows, a mobile-based OS
  - *Browser software*—Internet Explorer, Mozilla/Netscape, Opera, Safari, and others
  - *User interface components*—Active X, Java applets and others
  - *Plug-ins*—QuickTime, RealPlayer, and many others
  - *Connectivity*—cable, DSL, regular modem, T1
- Jumlah variabel konfigurasi harus dikurangi menjadi angka yang dapat dikelola

Security Testing

- Dirancang untuk memeriksa kerentanan lingkungan sisi klien, komunikasi jaringan yang terjadi ketika data diteruskan dari klien ke server dan kembali lagi, dan lingkungan sisi server
- Di sisi klien, kerentanan seringkali dapat ditelusuri ke bug yang sudah ada di browser, program email, atau perangkat lunak komunikasi.
- Di sisi server, kerentanan termasuk serangan penolakan layanan dan skrip berbahaya yang dapat diteruskan ke sisi klien atau digunakan untuk menonaktifkan operasi server

Performance Testing

- Apakah waktu respon server menurun ke titik di mana itu terlihat dan tidak dapat diterima?
- Pada titik apa (dalam hal pengguna, transaksi atau pemuatan data) apakah kinerja menjadi tidak dapat diterima?
- Komponen sistem apa yang bertanggung jawab atas penurunan kinerja?
- Berapa waktu respons rata-rata untuk pengguna dalam berbagai kondisi pemuatan?
- Apakah penurunan kinerja berdampak pada keamanan sistem?
- Apakah keandalan atau akurasi WebApp terpengaruh ketika beban pada sistem bertambah?
- Apa yang terjadi ketika beban yang lebih besar dari kapasitas server maksimum diterapkan?

Load Testing

- Tujuannya adalah untuk menentukan bagaimana WebApp dan lingkungan sisi servernya akan merespons berbagai kondisi loading
  - *N,* umlah pengguna bersamaan
  - *T,* jumlah transaksi online per unit waktu
  - *D,* beban data diproses oleh server per transaksi
- Throughput keseluruhan, P, dihitung dengan cara berikut:
  - $P = N \times T \times D$

Stress Testing

- Apakah sistem menurun dengan lembut atau apakah server dimatikan karena kapasitas terlampaui?
- Apakah perangkat lunak server menghasilkan pesan “server tidak tersedia”? Secara umum, apakah pengguna sadar bahwa mereka tidak dapat mencapai server?
- Apakah server antrian meminta sumber daya dan mengosongkan antrian begitu permintaan kapasitas berkurang?
- Apakah transaksi hilang karena kapasitas terlampaui?
- Apakah integritas data terpengaruh ketika kapasitas terlampaui?
- Nilai N, T, dan D apa yang menyebabkan lingkungan server gagal? Bagaimana kegagalan memanifestasikan dirinya? Apakah pemberitahuan otomatis dikirim ke staf dukungan teknis di situs server?
- Jika sistem gagal, berapa lama untuk kembali online?
- Apakah fungsi WebApp tertentu (mis., menghitung fungsionalitas intensif, kemampuan streaming data) dihentikan karena kapasitas mencapai tingkat 80 atau 90 persen?

Testing Mobile Applications

Mobile App Testing Strategy Questions

- Apakah Anda harus membuat prototipe yang berfungsi penuh sebelum Anda menguji dengan pengguna?
- Haruskah Anda menguji dengan perangkat pengguna atau menyediakan perangkat untuk pengujian?
- Perangkat dan grup pengguna apa yang harus Anda sertakan dalam pengujian?
- Kapan pengujian laboratorium versus pengujian jarak jauh sesuai?

Mobile Testing Guidelines

- Memahami lanskap jaringan dan lanskap perangkat.
- Melakukan pengujian dalam kondisi pengujian dunia nyata yang tidak terkendali.
- Pilih alat uji otomatisasi yang tepat.
- Identifikasi kombinasi perangkat keras / platform yang paling penting untuk diuji.
- Periksa aliran fungsional ujung-ke-ujung di semua platform yang mungkin setidaknya sekali.
- Melakukan pengujian kinerja, GUI, dan kompatibilitas menggunakan perangkat yang sebenarnya.
- Ukur kinerja MobileApp dalam kondisi beban jaringan yang realistis.

Mobile App Testing

- Conceptual Testing
- Unit and System Testing
- User Experience Testing
- Stability Testing
- Connectivity Testing
- Performance Testing
- Compatibility Testing
- Security Testing
- Certification Testing

Automated Testing

- Feasibility analysis
- Proof of concept
- Best practice test framework
- Customize testing tools
- Test under real world conditions
- Rapid defect resolution
- Reuse of test scripts

Stress Test Cases

- Menjalankan beberapa aplikasi seluler pada perangkat yang sama
- Menginfeksi perangkat lunak sistem dengan virus atau malware
- Mencoba mengambil alih perangkat dan menggunakannya untuk menyebarkan spam
- Memaksa aplikasi seluler untuk memproses sejumlah besar transaksi,
- Menyimpan sejumlah besar data pada perangkat

Mobile Usability Elements

- Fungsionalitas
- Arsitektur informasi
- Desain Layar
- Mekanisme input pengguna
- Konteks seluler diperhitungkan
- Kegunaan antarmuka
- Kepercayaan
- Umpan balik
- Fasilitas bantuan

Specialized Usability Tests

- Gestures
- Input dan pengenalan suara
- Input keyboard virtual
- Alerts dan errors

Mobile App Testing Tools

- Mobile page compliance checkers
- Mobile browser emulators
- Device emulators
- Key logging and playback
- Network monitors
- Mobile analytics collectors