**Jawaban Soal Test QA enginer**

1. Apa itu QA (Quality Assurance) dan apa peran utama seorang QA Engineer dalam siklus pengembangan perangkat lunak?

Jawaban:

Proses QA (Quality Assurance) bertujuan untuk memeriksa apakah perangkat lunak yang sedang dikembangkan telah memenuhi standar kualitas yang ditetapkan sebelum dapat digunakan oleh pengguna. QA tidak hanya menekankan pada identifikasi bug, tetapi juga memperhatikan peningkatan proses pengembangan guna menciptakan produk berkualitas tinggi yang sesuai dengan spesifikasi yang diinginkan.

Fungsi Utama Seorang QA Engineer dalam Proses Pengembangan Perangkat Lunak:

1. Membuat Rencana Pengujian:

QA Engineer memiliki tanggung jawab merencanakan strategi pengujian yang meliputi pengujian fungsional, non-fungsional, regresi, dan pengguna. Langkah ini menjamin semua komponen perangkat lunak akan diuji secara menyeluruh.

2. Membuat dan Menjalankan Test Case:

QA Engineer menulis test case berdasarkan spesifikasi produk dan memastikan bahwa semua fitur diuji dengan skenario yang mencerminkan penggunaan dunia nyata. Test case ini harus mencakup semua fungsionalitas perangkat lunak serta mengidentifikasi kemungkinan kesalahan.

3. Otomasi Pengujian:

Dalam beberapa kasus, QA Engineer harus membuat tes otomatis dengan menggunakan alat seperti Selenium, JUnit, atau TestNG untuk mempercepat pengujian regresi dan meningkatkan efisiensi uji coba.

4. Identifikasi dan Pelaporan Bug:

Salah satu peran utama QA Engineer adalah menemukan bug atau isu dalam perangkat lunak. Saat bug ditemukan, mereka mencatatnya dengan detail, termasuk cara-cara untuk mengulanginya, tingkat keparahan bug, dan pengaruhnya pada sistem.

5. Bekerja Sama dengan Tim Pengembang:

QA Engineer bekerjasama dengan tim pengembang untuk memahami fungsi-fungsi yang sedang dikembangkan, dan memberi masukan tentang kesalahan yang ditemukan. Kerjasama ini mempercepat peningkatan kualitas produk dengan memperbaiki bug.

6. Melakukan Pengujian Ulang (Retesting):

Setelah bug diperbaiki oleh pengembang, QA Engineer melakukan pengujian ulang untuk memastikan bahwa tidak ada bug baru yang muncul setelah masalah tersebut terselesaikan.

7. Pengujian Regressi:

QA Engineer juga melakukan pengujian regresi, yaitu menguji ulang bagian dari perangkat lunak yang tidak terkait langsung dengan perubahan terbaru untuk memastikan bahwa perubahan tersebut tidak menyebabkan masalah di area lain.

8. Verifikasi Kualitas Akhir:

Sebelum perangkat lunak dirilis, QA Engineer bertugas untuk memastikan bahwa semua pengujian telah dilaksanakan dengan baik, dan produk yang akan dirilis sudah sesuai dengan standar kualitas yang diinginkan.

1. Jelaskan perbedaan antara pengujian fungsional dan pengujian non-fungsional dalam konteks QA.

Jawaban:

Pengujian fungsional adalah pengujian yang memerhatikan fungsi dan fitur perangkat lunak untuk memastikan kesesuaian sistem dengan spesifikasi yang ditetapkan. Sedangkan Pengujian non-fungsional berpusat pada kinerja, keandalan, dan pengalaman pengguna dari suatu perangkat lunak, bukan sekadar memeriksa apakah fungsi tertentu berjalan dengan baik.

**Perbedaan Utama:**

* Fokus:
  + Pengujian Fungsional: Fokus pada aktivitas yang dilakukan sistem (fungsi atau fitur).
  + Pengujian Non-Fungsional: Fokus pada pada cara kerja sistem (kinerja, keamanan, keandalan).
* Pendekatan:
  + Pengujian Fungsional: Didasarkan pada spesifikasi dan kebutuhan fungsional.
  + Pengujian Non-Fungsional: Melibatkan evaluasi metrik kinerja dan aspek-aspek non-fungsional dari software.
* Tujuan:
  + Pengujian Fungsional: Bertujuan untuk memeriksa apakah fungsi-fungsi individu beroperasi sesuai dengan yang telah ditentukan dalam spesifikasi.
  + Pengujian Non-Fungsional: Bertujuan untuk menjamin kualitas keseluruhan perangkat lunak meliputi kecepatan, keamanan, skalabilitas, dan lainnya.

1. Bagaimana Anda akan merencanakan dan melakukan pengujian manual untuk aplikasi web e-commerce? Berikan langkah-langkah utama dalam proses ini.

Jawaban:

Melakukan pengujian manual untuk aplikasi web e-commerce memerlukan strategi yang terstruktur guna menjamin bahwa semua fitur aplikasi berjalan sesuai yang diinginkan. Inilah tahapan penting dalam proses pengujian secara manual:

1. Mengerti Kriteria dan Kebutuhan Aplikasi

Tahap awal adalah memahami semua spesifikasi fungsional dan non-fungsional dari aplikasi e-commerce yang akan diuji, yang termasuk:

- Fitur utama (misalnya, pencarian produk, keranjang belanja, pembayaran).

- Integrasi dengan sistem pembayaran.

- Proses bisnis yang penting (membeli, mengirimkan, mengelola barang).

- Sasaran pengguna (contohnya, apakah pengguna menggunakan aplikasi di perangkat mobile atau desktop).

2. Menyusun Rencana Pengujian (Test Plan)

Dokumen pengujian adalah detail strategi pengujian. Beberapa komponen yang perlu terdapat dalam rencana pengujian yaitu:

- Tujuan Pengujian: Tujuan yang hendak dicapai dalam pengujian ini seperti mengenal pasti kesilapan, mengesahkan ciri-ciri.

- Jangkauan Pengujian: Fitur-fitur apa yang akan diuji (seperti halaman produk, fitur checkout, manajemen akun).

- Sumber Daya: Siapa yang akan melaksanakan pengujian dan peralatan apa yang dibutuhkan.

- Jadwal Testing: Kapan Testing akan dijalankan dan berapa durasi setiap Testing.

3. Membuat Test Case

Test case adalah skenario pengujian yang spesifik, biasanya dibuat berdasarkan persyaratan dan spesifikasi aplikasi. Setiap test case harus memuat:

- Tahapan pengujian yaitu panduan yang harus dipatuhi saat melakukan pengujian.

- Data atau langkah-langkah yang perlu diambil untuk melakukan pengujian.

- Expected result yaitu hasil yang diharapkan dari pengujian tersebut.

Test case ini harus mencakup semua fungsi utama dari aplikasi web e-commerce seperti:

- Pendaftaran akun

- Pencarian produk

- Pengelolaan keranjang belanja

- Pembayaran dan checkout

- Layanan pelanggan, dll

4. Melakukan Pengujian Fungsional

Setelah pembuatan test case selesai, dapat dimulai melakukan pengujian fungsional. Di dalam fase ini ada beberapa hal yang penting:

- Melakukan setiap uji coba dengan mengikuti instruksi yang sudah tertera.

- Memasukkan berbagai skenario dan input yang valid maupun invalid.

- Memastikan bahwa setiap fitur aplikasi bekerja sesuai yang diharapkan.

5. Melakukan Pengujian Non-Fungsional

Pengujian non-fungsional dilakukan untuk memastikan bahwa aplikasi dapat menangani berbagai skenario seperti:

- Performance Testing: Menguji kecepatan respons aplikasi saat diakses oleh banyak pengguna sekaligus.

- Security Testing: Memastikan bahwa transaksi aman, data pengguna terlindungi, dan tidak ada celah keamanan.

- Pengujian Usability: Mengevaluasi apakah pengguna dapat dengan mudah menavigasi aplikasi dan menyelesaikan transaksi tanpa hambatan.

6. Melaporkan Bug

Apabila menemukan bug atau masalah saat pengujian, catatlah detail lengkap bug tersebut, termasuk:

- Langkah-langkah untuk mereproduksi bug

- Lingkungan pengujian, (misalnya, browser yang digunakan, sistem operasi)

- Severity (tingkat keparahan), apakah bug ini kritis, besar, atau minor

7. Pengujian Ulang dan Pengujian Regresi

Setelah tim pengembang memperbaiki bug, dilakukan pengujian ulang untuk memastikan bahwa bug telah diatasi. Selain itu, dilakukan pengujian regresi untuk memastikan bahwa perubahan kode tidak menyebabkan bug baru di bagian lain dari aplikasi.

9. Mengkomunikasikan Hasil Pengujian

Setelah semua pengujian selesai, kemudian membuat laporan yang merinci mengenai:

- Fitur yang telah diuji

- Bug yang ditemukan dan statusnya

- Fitur yang lulus atau gagal dalam pengujian

1. Apa yang dimaksud dengan uji regresi? Mengapa uji regresi penting dalam pengujian perangkat lunak?

Jawaban:

Uji regresi adalah proses pengujian perangkat lunak yang dilakukan untuk memastikan bahwa perubahan atau perbaikan yang dilakukan pada kode tidak menyebabkan masalah baru atau memunculkan bug di bagian lain dari perangkat lunak yang sebelumnya sudah berfungsi dengan baik.

Mengapa Uji Regresi Penting dalam Pengujian Perangkat Lunak?

* **Menjaga Keandalan Sistem**

Tiap kali kode diubah, terdapat kemungkinan bahwa perubahan tersebut dapat mempengaruhi kinerja fitur yang telah ada. Uji regresi membantu menjaga stabilitas sistem secara keseluruhan dengan memeriksa bahwa perubahan kode tidak menimbulkan masalah baru.

* **Mendeteksi Bug Tak Terduga**

Modifikasi pada satu komponen sistem bisa berdampak pada bagian lain yang tidak terduga. Uji regresi dapat menemukan bug di area yang tidak terkait langsung dengan perubahan kode.

* **Menjamin Kualitas Perangkat Lunak**

Melalui uji regresi, QA memeriksa bahwa perangkat lunak tetap stabil dan beroperasi normal meskipun ada perubahan pada kode. Menjaga kualitas perangkat lunak secara menyeluruh adalah hal yang sangat penting.

* **Meningkatkan Kepercayaan Sebelum Rilis**

Uji regresi menunjukkan keyakinan bahwa perangkat lunak bisa diluncurkan atau diperbarui tanpa menimbulkan masalah bagi pengguna akhir. Hal ini menjamin bahwa perangkat lunak telah siap digunakan tanpa adanya penurunan fungsi.

* **Mengurangi Risiko Kesalahan Produksi**

Tanpa dilakukannya uji regresi, bisa saja bug baru tidak teridentifikasi dan muncul setelah perangkat lunak dilepas, menyebabkan kerugian pada waktu, biaya, dan reputasi perusahaan. Uji regresi membantu mengurangi risiko ini dengan menemukan bug sebelum perangkat lunak dilepaskan ke produksi.

1. Apa perbedaan antara pengujian otomatis dan pengujian manual? Kapan Anda harus menggunakan pengujian otomatis daripada pengujian manual?

Jawaban:

Pengujian manual terjadi ketika seorang penguji melakukan pengujian perangkat lunak dengan menjalankan test case tanpa menggunakan alat otomatisasi. Penguji mematuhi skenario pengujian yang telah disiapkan dan memperhatikan respons perangkat lunak terhadap input spesifik.

Pengujian secara otomatis merupakan teknik pengujian yang melibatkan eksekusi test case secara otomatis dengan menggunakan perangkat skrip dan alat khusus seperti Selenium, JUnit, atau TestNG. Pengujian sering dimanfaatkan untuk pengujian berulang dengan waktu yang lebih efisien dan cepat.

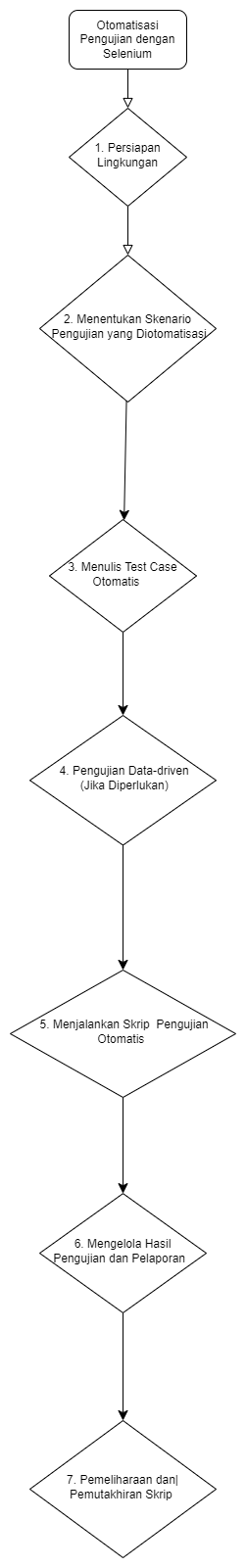
**Kapan Menggunakan Pengujian Otomatis Daripada Pengujian Manual:**

Pengujian otomatis dilakukan saat:

* Pengujian Regresi: Ketika ada perubahan kode yang sering terjadi dan perlu memastikan bahwa fungsionalitas lama tetap bekerja dengan baik.
* Pengujian Kinerja: Untuk mengukur waktu respons atau memuat beban pengguna dalam jumlah besar yang sulit atau tidak mungkin dilakukan secara manual.
* Pengujian Berulang: Ketika fitur tertentu harus diuji berulang kali dengan input yang berbeda, seperti pengujian berbagai kombinasi metode pembayaran di aplikasi e-commerce.
* Pengujian yang Memerlukan Hasil Cepat: Dalam proses continuous integration atau continuous deployment (CI/CD), pengujian otomatis memungkinkan pengujian yang lebih cepat dan lebih sering.

1. Gambarkan aliran kerja atau proses yang akan Anda ikuti untuk mengotomatisasi pengujian menggunakan alat seperti Selenium.

Jawaban:



1. Apa itu kerangka kerja pengujian (testing framework) dalam pengujian otomatis, dan bagaimana kerangka kerja ini membantu dalam pengujian perangkat lunak?

Jawaban:  
Testing Framework adalah kumpulan aturan, pedoman, praktik, dan alat yang terstruktur untuk mendukung proses pengujian otomatis dalam pengembangan perangkat lunak. Framework ini berfungsi untuk mempermudah penulisan, eksekusi, dan manajemen pengujian otomatis secara terstruktur, efektif, dan mudah untuk dikelola.

**Bagaimana Kerangka Kerja Membantu dalam Pengujian Perangkat Lunak:**

* Otomatisasi yang Efisien

Kerangka kerja meningkatkan struktur dan efisiensi pengujian otomatis. Pengujian dapat diulang tanpa campur tangan manual secara konsisten, sehingga akurasi meningkat.

* Peningkatan Skalabilitas

Dengan menggunakan framework, pengujian otomatis dapat diperluas secara mudah untuk menjangkau skenario-skenario yang lebih kompleks seiring dengan pertumbuhan aplikasi.

* Kolaborasi yang Lebih Baik

Konsep seperti BDD memungkinkan tim bisnis, QA, dan pengembang untuk bekerjasama dengan pemahaman yang seragam tentang pengujian.

* Pelacakan Bug yang Lebih Mudah

Dengan bantuan laporan otomatis dari framework seperti TestNG atau JUnit, pengembang dan QA dapat dengan mudah menemukan dan mengatasi bug berdasarkan hasil pengujian yang akurat.

1. Apa yang dimaksud dengan "bug tracking system" dan sebutkan beberapa alat umum yang digunakan untuk melacak dan mengelola bug dalam perangkat lunak.

Jawaban:

Sistem Pelacakan Bug adalah sebuah sistem atau perangkat yang dipakai untuk memantau, mengatur, dan menyusun laporan tentang bug atau masalah dalam perangkat lunak selama tahap pengembangan dan pengujian. Sistem ini membantu tim pengembang dan Quality Assurance (QA) dalam mendokumentasikan, menetapkan prioritas, melacak status, dan memantau penyelesaian masalah atau bug yang telah ditemukan.

**Alat-Alat Umum yang Digunakan untuk Melacak dan Mengelola Bug:**

* JIRA

JIRA adalah salah satu alat manajemen proyek dan pelacakan bug yang paling populer. Alat ini mendukung pelacakan bug, manajemen tugas, dan juga digunakan untuk implementasi metode Agile dan Scrum.

* Bugzilla

Bugzilla adalah alat pelacakan bug open-source yang dikelola oleh Mozilla. Ini sangat populer di komunitas pengembang karena kesederhanaannya.

* MantisBT

MantisBT adalah alat pelacakan bug berbasis web yang bersifat open-source, dirancang untuk memberikan kemudahan penggunaan sekaligus fleksibilitas bagi penggunanya.

* Redmine

Redmine adalah aplikasi manajemen proyek berbasis web dengan dukungan untuk pelacakan bug. Alat ini bersifat open-source dan mendukung multiple projects serta multiple users.

* Trello

Trello adalah alat manajemen proyek berbasis kanban yang sering digunakan untuk pelacakan bug secara sederhana. Alat ini memungkinkan tim untuk membuat "card" untuk setiap bug yang dapat dipindahkan di antara kolom status.

* Asana

Asana adalah alat manajemen proyek yang dapat digunakan untuk melacak bug bersama tugas-tugas lain dalam proyek. Asana banyak digunakan oleh tim yang menginginkan pendekatan kolaboratif.

* GitHub Issues

GitHub Issues adalah alat pelacakan bug yang terintegrasi langsung dengan repositori GitHub, yang memungkinkan pengembang melaporkan dan melacak masalah secara langsung pada proyek open-source atau private repository.

* YouTrack

YouTrack adalah alat pelacakan masalah dan bug yang dikembangkan oleh JetBrains. Alat ini sering digunakan oleh tim pengembangan yang membutuhkan fleksibilitas tinggi dalam pelacakan masalah.

1. Berikan contoh skenario uji fungsional untuk aplikasi pemesanan tiket pesawat secara online.

Jawaban:

1. Skenario Pencarian Penerbangan

- Tujuan Pengujian: Memastikan pengguna dapat mencari penerbangan berdasarkan kriteria yang dimasukkan.

- Langkah-langkah Pengujian:

1. Buka halaman utama aplikasi pemesanan tiket.

2. Masukkan kota asal (misal: Jakarta) dan kota tujuan (misal: Bali).

3. Pilih tanggal keberangkatan dan tanggal kembali (jika ada).

4. Pilih jumlah penumpang (dewasa, anak-anak, bayi).

5. Klik tombol "Cari Penerbangan".

- Hasil yang Diharapkan:

- Sistem menampilkan daftar penerbangan yang sesuai dengan kriteria yang dimasukkan.

- Informasi yang muncul mencakup harga tiket, waktu keberangkatan dan kedatangan, maskapai penerbangan, dan durasi penerbangan.

2. Skenario Pemesanan Tiket

- Tujuan Pengujian: Memastikan pengguna dapat memesan tiket dengan benar.

- Langkah-langkah Pengujian:

1. Pilih penerbangan dari hasil pencarian yang ditampilkan.

2. Klik tombol "Pesan Sekarang".

3. Masukkan detail penumpang, seperti nama lengkap, jenis kelamin, dan nomor KTP.

4. Pilih tempat duduk (jika fitur tersedia).

5. Klik "Lanjutkan" untuk melanjutkan ke metode pembayaran.

- Hasil yang Diharapkan:

- Sistem menampilkan ringkasan pemesanan yang benar dan memberikan opsi untuk melanjutkan ke pembayaran.

- Informasi detail penerbangan, jumlah penumpang, dan total biaya ditampilkan dengan akurat.

3. Skenario Pembayaran Tiket

- Tujuan Pengujian: Memastikan pengguna dapat melakukan pembayaran dengan metode yang disediakan.

- Langkah-langkah Pengujian:

1. Pilih metode pembayaran (misal: kartu kredit, transfer bank, atau dompet digital).

2. Masukkan detail pembayaran (misal: nomor kartu kredit dan CVV untuk pembayaran kartu kredit).

3. Klik tombol "Bayar Sekarang".

- Hasil yang Diharapkan:

- Sistem memproses pembayaran dan memberikan konfirmasi.

- Pengguna menerima email konfirmasi pemesanan yang mencakup e-ticket atau rincian penerbangan.

4. Skenario Pembatalan Pemesanan

- Tujuan Pengujian: Memastikan pengguna dapat membatalkan pemesanan tiket.

- Langkah-langkah Pengujian:

1. Buka halaman "Pemesanan Saya".

2. Pilih pemesanan yang ingin dibatalkan.

3. Klik tombol "Batalkan Pemesanan".

4. Konfirmasi pembatalan.

- Hasil yang Diharapkan:

- Sistem membatalkan pemesanan dan mengirimkan email konfirmasi pembatalan kepada pengguna.

- Jika ada kebijakan pengembalian uang, pengguna menerima pemberitahuan mengenai proses refund.

5. Skenario: Memeriksa Status Penerbangan

- Tujuan Pengujian: Memastikan pengguna dapat memeriksa status penerbangan secara real-time.

- Langkah-langkah Pengujian:

1. Buka halaman "Status Penerbangan".

2. Masukkan nomor penerbangan atau rute penerbangan (misal: Jakarta ke Bali).

3. Klik tombol "Cek Status".

- Hasil yang Diharapkan:

- Sistem menampilkan status penerbangan terkini, seperti "On Time", "Delayed", atau "Cancelled".

- Informasi tentang gate keberangkatan dan waktu boarding ditampilkan jika tersedia.

6. Skenario Penambahan Bagasi Tambahan

- Tujuan Pengujian: Memastikan pengguna dapat menambah bagasi setelah memesan tiket.

- Langkah-langkah Pengujian:

1. Buka halaman "Pemesanan Saya".

2. Pilih pemesanan yang aktif.

3. Klik tombol "Tambahkan Bagasi".

4. Pilih jumlah tambahan bagasi dan klik "Konfirmasi".

- Hasil yang Diharapkan:

- Sistem menampilkan jumlah biaya tambahan untuk bagasi dan memproses pembayarannya.

- Pengguna menerima email konfirmasi dengan rincian bagasi tambahan.

7. Skenario Pengubahan Tanggal atau Rute Penerbangan

- Tujuan Pengujian: Memastikan pengguna dapat mengubah tanggal atau rute penerbangan jika diperlukan.

- Langkah-langkah Pengujian:

1. Buka halaman "Pemesanan Saya".

2. Pilih pemesanan yang ingin diubah.

3. Klik tombol "Ubah Penerbangan".

4. Pilih tanggal atau rute penerbangan baru.

5. Klik "Konfirmasi Perubahan".

- Hasil yang Diharapkan:

- Sistem menampilkan perubahan yang berhasil, termasuk biaya perubahan jika ada.

- Pengguna menerima email konfirmasi perubahan penerbangan.

8. Skenario Penggunaan Kode Promo atau Diskon

- Tujuan Pengujian: Memastikan pengguna dapat menggunakan kode promo atau diskon saat memesan tiket.

- Langkah-langkah Pengujian:

1. Saat melakukan pemesanan, masukkan kode promo di kolom yang disediakan.

2. Klik tombol "Terapkan Kode".

- Hasil yang Diharapkan:

- Sistem mengurangi total biaya sesuai dengan diskon dari kode promo yang valid.

- Pengguna dapat melanjutkan proses pembayaran dengan biaya yang telah didiskon.

9. Skenario Registrasi dan Login Pengguna

- Tujuan Pengujian: Memastikan pengguna dapat membuat akun dan login ke aplikasi.

- Langkah-langkah Pengujian:

1. Buka halaman registrasi.

2. Masukkan data pengguna (nama, email, nomor telepon, kata sandi).

3. Klik tombol "Daftar".

4. Setelah registrasi, coba login menggunakan akun yang baru dibuat.

- Hasil yang Diharapkan:

- Pengguna berhasil membuat akun dan login ke aplikasi tanpa masalah.

- Setelah login, pengguna dapat melihat dan mengelola pemesanan mereka.

1. Anda sedang menguji aplikasi perbankan online. Bagaimana Anda akan menguji keamanan aplikasi ini? Berikan beberapa contoh uji keamanan yang akan Anda lakukan.

Jawaban:

* + **Pengujian Otentikasi (Authentication Testing)**

Memastikan bahwa hanya pengguna yang sah yang dapat mengakses akun perbankan mereka.

Contoh Uji:

Menguji kekuatan kata sandi: Pastikan aplikasi memerlukan kata sandi yang kuat dengan kombinasi huruf besar, huruf kecil, angka, dan simbol.

* **Pengujian Otorisasi (Authorization Testing)**

Memastikan bahwa pengguna hanya memiliki akses ke fungsi dan data yang sesuai dengan hak akses mereka.

Contoh Uji:

Mencoba akses halaman atau fungsi yang seharusnya tidak dapat diakses oleh pengguna dengan hak akses rendah (misalnya, akses ke data pengguna lain atau fitur administratif).

* **Pengujian Kerentanan SQL Injection**

Tidak rentan terhadap serangan **SQL Injection**, di mana penyerang dapat memasukkan kode SQL berbahaya ke dalam input aplikasi untuk mendapatkan akses tidak sah ke basis data.

* **Pengujian Enkripsi Data**

Memastikan bahwa data sensitif seperti informasi login dan data transaksi dienkripsi dengan benar baik saat transit maupun saat disimpan.

Contoh Uji:

Melakukan cek apakah data yang dikirim antara klien dan server (misalnya informasi login atau detail transaksi) dienkripsi menggunakan protokol **SSL/TLS** (misal, https).

* **Pengujian Keamanan API**

Memastikan bahwa API yang digunakan oleh aplikasi perbankan terlindungi dari serangan.

Contoh Uji:

Menguji apakah API memiliki autentikasi yang memadai, seperti token OAuth atau API key.

* **Pengujian Log Aktivitas Pengguna**

Memastikan bahwa semua aktivitas penting pengguna dilog dengan benar sehingga dapat diaudit jika terjadi insiden keamanan.

Contoh Uji:

Melakukan cek apakah aplikasi melog semua tindakan penting seperti login, logout, perubahan kata sandi, transfer dana, dan aktivitas sensitif lainnya.

1. Pada Aplikasi Mobile Shopee, pada Fitur Pesanan Saya - Tab Belum Bayar terdapat penambahan aturan terkait Edit Jenis Pembayaran hanya bisa dilakukan pada kondisi order : a) Jika Order tsb belum dibayar

b) Jenis Pembayaran yg sudah dipilih memiliki status ‘gagal-bayar’

Jika salah satu syarat tsb terpenuhi maka pada Fitur Pesanan Saya- TabBelum Bayar pada baris order nya terdapat button ‘Ubah Pembayaran’. Coba Buatkan Test Case terkait update fitur tsb dengan menggunakanTest Variable serta contoh Test Data berdasarkan ketentuan diatas.

Jawaban:

Test Case: Ubah Pembayaran pada Fitur "Pesanan Saya" - Tab Belum Bayar

Tujuan Pengujian:

Memastikan bahwa tombol "Ubah Pembayaran" hanya muncul pada order di tab "Belum Bayar" ketika:

1. Order belum dibayar.

2. Jenis pembayaran yang dipilih memiliki status gagal bayar.

Test Variables:

1. Status Pembayaran Order:

- Belum Dibayar

- Dibayar

- Gagal Bayar

2. Status Jenis Pembayaran:

- Pembayaran Berhasil

- Pembayaran Gagal

- Pembayaran Menunggu

3. Jenis Pembayaran yang Dipilih:

- Transfer Bank

- E-Wallet

- Kartu Kredit

Test Case 1: Tombol "Ubah Pembayaran" Muncul Ketika Order Belum Dibayar

- ID Test Case: TC001

- Skenario Pengujian: Memastikan tombol "Ubah Pembayaran" muncul jika status order belum dibayar.

- Langkah Pengujian:

1. Lakukan order produk di Shopee.

2. Pilih jenis pembayaran "Transfer Bank".

3. Jangan lakukan pembayaran.

4. Akses fitur "Pesanan Saya" - Tab Belum Bayar.

- Test Data:

- Status Pembayaran Order: Belum Dibayar

- Jenis Pembayaran yang Dipilih: Transfer Bank

- Hasil yang Diharapkan: Tombol "Ubah Pembayaran" muncul di baris order tersebut.

Test Case 2: Tombol "Ubah Pembayaran" Tidak Muncul Jika Order Sudah Dibayar

- ID Test Case: TC002

- Skenario Pengujian: Memastikan tombol "Ubah Pembayaran" tidak muncul jika status order sudah dibayar.

- Langkah Pengujian:

1. Lakukan order produk di Shopee.

2. Pilih jenis pembayaran "E-Wallet".

3. Lakukan pembayaran dengan sukses.

4. Akses fitur "Pesanan Saya" - Tab Belum Bayar.

- Test Data:

- Status Pembayaran Order: Dibayar

- Jenis Pembayaran yang Dipilih: E-Wallet

- Hasil yang Diharapkan: Tombol "Ubah Pembayaran" tidak muncul di baris order tersebut.

Test Case 3: Tombol "Ubah Pembayaran" Muncul Jika Jenis Pembayaran Gagal Bayar

- ID Test Case: TC003

- Skenario Pengujian: Memastikan tombol "Ubah Pembayaran" muncul jika jenis pembayaran yang dipilih memiliki status gagal bayar.

- Langkah Pengujian:

1. Lakukan order produk di Shopee.

2. Pilih jenis pembayaran "Kartu Kredit".

3. Simulasikan kegagalan pembayaran (misalnya karena otorisasi gagal).

4. Akses fitur "Pesanan Saya" - Tab Belum Bayar.

- Test Data:

- Status Pembayaran Order: Belum Dibayar

- Status Jenis Pembayaran: Gagal Bayar

- Jenis Pembayaran yang Dipilih: Kartu Kredit

- Hasil yang Diharapkan: Tombol "Ubah Pembayaran" muncul di baris order tersebut.

Test Case 4: Tombol "Ubah Pembayaran" Tidak Muncul Jika Pembayaran Berhasil

- ID Test Case: TC004

- Skenario Pengujian: Memastikan tombol "Ubah Pembayaran" tidak muncul jika pembayaran sudah berhasil.

- Langkah Pengujian:

1. Lakukan order produk di Shopee.

2. Pilih jenis pembayaran "E-Wallet".

3. Lakukan pembayaran dengan sukses.

4. Akses fitur "Pesanan Saya" - Tab Belum Bayar.

- Test Data:

- Status Pembayaran Order: Dibayar

- Status Jenis Pembayaran: Pembayaran Berhasil

- Jenis Pembayaran yang Dipilih: E-Wallet

- Hasil yang Diharapkan: Tombol "Ubah Pembayaran" tidak muncul di baris order tersebut.

Test Case 5: Tombol "Ubah Pembayaran" Muncul Jika Order Belum Dibayar dan Status Pembayaran Menunggu

- ID Test Case: TC005

- Skenario Pengujian: Memastikan tombol "Ubah Pembayaran" muncul jika order belum dibayar dan status pembayaran masih menunggu.

- Langkah Pengujian:

1. Lakukan order produk di Shopee.

2. Pilih jenis pembayaran "Transfer Bank".

3. Jangan lakukan pembayaran, sehingga statusnya masih menunggu.

4. Akses fitur "Pesanan Saya" - Tab Belum Bayar.

- Test Data:

- Status Pembayaran Order: Belum Dibayar

- Status Jenis Pembayaran: Menunggu

- Jenis Pembayaran yang Dipilih: Transfer Bank

- Hasil yang Diharapkan: Tombol "Ubah Pembayaran" muncul di baris order tersebut.

Test Case 6: Validasi Fungsi Tombol "Ubah Pembayaran"

- ID Test Case: TC006

- Skenario Pengujian: Memastikan tombol "Ubah Pembayaran" dapat digunakan untuk mengubah jenis pembayaran.

- Langkah Pengujian:

1. Akses fitur "Pesanan Saya" - Tab Belum Bayar.

2. Pilih order yang memiliki status "Belum Dibayar".

3. Klik tombol "Ubah Pembayaran".

4. Pilih metode pembayaran baru (misalnya dari "Transfer Bank" menjadi "E-Wallet").

5. Lakukan pembayaran dengan metode yang baru dipilih.

- Test Data:

- Status Pembayaran Order: Belum Dibayar

- Jenis Pembayaran Awal: Transfer Bank

- Jenis Pembayaran Baru: E-Wallet

- Hasil yang Diharapkan: Pembayaran berhasil dilakukan menggunakan metode baru, dan order berpindah dari tab "Belum Bayar" ke "Dikemas".