**Agile Testing**

1. Quality Engineer (QE) berperan dalam berbagai fase proses Software Development Life Cycle (SDLC) untuk memastikan kualitas perangkat lunak yang dikembangkan. Berikut adalah beberapa fase dalam SDLC dan peran yang QE dapat lakukan pada setiap fase tersebut:
2. Requirements Gathering (Pengumpulan Persyaratan):

QE dapat berperan dalam memastikan bahwa persyaratan yang dikumpulkan adalah jelas, terukur, dan dapat diuji. QE dapat berkolaborasi dengan tim bisnis dan pemangku kepentingan untuk memahami persyaratan secara mendalam, mengajukan pertanyaan, dan mengidentifikasi aspek kualitas yang perlu diperhatikan.

1. Design (Perancangan):

Pada fase perancangan, QE dapat melakukan review terhadap desain perangkat lunak yang diajukan. QE dapat mengidentifikasi potensi masalah kualitas, seperti desain yang sulit untuk diuji atau desain yang tidak memenuhi standar atau praktik terbaik dalam pengembangan perangkat lunak. QE juga dapat memberikan masukan untuk meningkatkan desain agar lebih mudah diuji, lebih skalabel, atau lebih aman.

1. Development (Pengembangan):

Selama fase pengembangan, QE dapat berperan dalam pengujian unit (unit testing) untuk memastikan bahwa komponen individu perangkat lunak berfungsi dengan baik dan sesuai dengan spesifikasi. QE juga dapat membantu mengidentifikasi dan menghapus bug atau masalah dalam kode.

1. Testing (Pengujian):

Ini adalah fase di mana QE memiliki peran utama. QE bertanggung jawab untuk merencanakan, merancang, dan melaksanakan strategi pengujian yang komprehensif. QE dapat melibatkan pengujian fungsional, pengujian integrasi, pengujian kinerja, pengujian keamanan, dan pengujian lainnya. QE juga dapat melakukan otomatisasi pengujian untuk meningkatkan efisiensi dan akurasi pengujian. QE dapat melaporkan bug atau masalah yang ditemukan kepada tim pengembangan, memberikan pemahaman mendalam tentang kualitas perangkat lunak yang sedang diuji.

1. Deployment (Implementasi):

QE dapat melakukan pengujian akhir sebelum implementasi untuk memastikan bahwa perangkat lunak siap untuk diperkenalkan ke lingkungan produksi. QE dapat memvalidasi bahwa perangkat lunak berjalan dengan baik pada infrastruktur yang dituju, memastikan ketersediaan dan skalabilitas, serta mengidentifikasi potensi risiko dan dampaknya terhadap pengguna.

1. Maintenance (Pemeliharaan):

Setelah perangkat lunak dideploy, QE dapat terlibat dalam pemeliharaan perangkat lunak dengan melakukan pengujian regresi untuk memastikan bahwa pembaruan atau perbaikan tidak mempengaruhi fungsionalitas yang ada. QE juga dapat memantau performa perangkat lunak dan mengidentifikasi dan menangani masalah yang timbul selama masa pemeliharaan.

Peran QE dalam setiap fase SDLC berfokus pada memastikan kualitas perangkat lunak dengan melakukan pengujian, memberikan umpan balik, dan berkolaborasi dengan tim pengembangan untuk meningkatkan kualitas.

1. Tantangan dalam menerapkan Agile Testing meliputi:
2. Waktu Terbatas: Agile mengikuti siklus pengembangan yang cepat, dengan iterasi yang singkat. Tantangan utama adalah menjalankan pengujian dalam waktu yang terbatas untuk memberikan umpan balik yang cepat dan mengatasi risiko kualitas.
3. Perubahan yang Cepat: Agile menekankan fleksibilitas dan respons terhadap perubahan. Tantangan adalah menghadapi perubahan persyaratan atau prioritas yang dapat terjadi setiap iterasi dan mengadaptasi skenario pengujian dengan cepat.
4. Kolaborasi Intensif: Manifesto Agile menekankan kolaborasi yang erat antara anggota tim. Tantangan adalah berkolaborasi secara intensif dengan tim pengembangan, berdiskusi tentang persyaratan, desain, dan penerjemahan ke dalam skenario pengujian yang relevan.
5. Pengelolaan Sumber Daya yang Terbatas: Dalam konteks Agile, prioritas pengujian harus dikelola dengan sumber daya yang terbatas. Tantangan adalah melakukan penilaian risiko yang efektif, fokus pada pengujian yang kritis, dan menggunakan sumber daya secara optimal.
6. Perbaikan Berkelanjutan: Agile mendorong perbaikan berkelanjutan dalam pengembangan perangkat lunak. Tantangan adalah melakukan evaluasi dan retrospektif untuk mengidentifikasi cara-cara baru dalam meningkatkan proses pengujian, meningkatkan otomatisasi, dan memperkuat kerja tim.
7. Keterlibatan Pelanggan: Agile menekankan kolaborasi yang erat dengan pelanggan. Tantangan adalah memastikan keterlibatan pelanggan dalam pengujian dan mendapatkan umpan balik yang berharga untuk memastikan bahwa perangkat lunak yang dikembangkan memenuhi kebutuhan pelanggan.
8. Pengujian Continous Integration dan Delivery: Agile mendorong praktik CI/CD yang melibatkan pengujian terus-menerus. Tantangan adalah mengotomatisasi pengujian, menjalankan pengujian unit, integrasi, dan penerimaan dengan cepat dan efisien dalam alur kerja CI/CD.
9. Dokumentasi yang Tepat: Meskipun Agile menekankan pengiriman perangkat lunak yang berfungsi lebih daripada dokumentasi yang komprehensif, tantangan adalah menciptakan dokumentasi yang relevan dan cukup untuk memahami pengujian yang telah dilakukan dan hasilnya.

Berikut adalah penjelasan pada masing-masing poin dalam Agile Manifesto terkait pengujian perangkat lunak:

1. Testing is an activity not a phase (Pengujian adalah aktivitas, bukan fase):

Poin ini menekankan pentingnya melibatkan pengujian sebagai aktivitas yang dilakukan secara terus-menerus sepanjang siklus pengembangan, bukan hanya sebagai fase terpisah. Pengujian tidak seharusnya terjadi hanya pada tahap akhir, tetapi dilakukan sepanjang waktu untuk memastikan kualitas perangkat lunak yang sedang dikembangkan.

1. Prevent bugs rather than finding bugs (Mencegah bug dari pada menemukan bug):

Poin ini menggaris bawahi pentingnya mencegah terjadinya bug sejak awal daripada hanya mencoba menemukan bug setelah perangkat lunak selesai dikembangkan. Tim pengembangan harus berfokus pada penggunaan praktik pengembangan yang baik dan pengujian yang terus-menerus untuk mengurangi kemungkinan adanya bug.

1. Don't be a checker, be a tester (Jangan menjadi pemeriksa, menjadi pengujian):

Poin ini menekankan pentingnya peran seorang pengujian dalam memberikan umpan balik berharga kepada tim pengembangan. Seorang pengujian tidak hanya melaksanakan tugas pemeriksaan secara mekanis, tetapi juga harus memahami konteks bisnis dan tujuan perangkat lunak untuk menguji dengan sudut pandang yang lebih luas.

1. Don't try to break the system, instead help build the best possible system (Jangan mencoba merusak sistem, sebaliknya membantu membangun sistem yang terbaik):

Poin ini menekankan bahwa peran pengujian tidak hanya berfokus pada menemukan kelemahan atau bug dalam sistem, tetapi juga berkontribusi dalam membangun sistem yang terbaik. Tim pengujian harus berkolaborasi dengan tim pengembangan untuk memastikan bahwa perangkat lunak yang dikembangkan mencapai kualitas yang diinginkan.

1. The whole team is responsible for quality, not just the tester (Seluruh tim bertanggung jawab terhadap kualitas, bukan hanya pengujian):

Poin ini menekankan bahwa tanggung jawab untuk mencapai kualitas perangkat lunak bukan hanya menjadi tanggung jawab seorang pengujian, tetapi menjadi tanggung jawab seluruh tim pengembangan. Setiap anggota tim harus berpartisipasi dalam upaya untuk memastikan kualitas perangkat lunak, termasuk melibatkan diri dalam pengujian dan memprioritaskan kualitas selama seluruh siklus pengembangan.

Dengan memahami dan menerapkan prinsip-prinsip ini, tim pengembangan dapat mencapai pengujian yang lebih efektif dan terintegrasi dalam proses pengembangan perangkat lunak secara keseluruhan.