Pertemuan 11

Pengujian Regresi (Regression Testing)

Deskripsi Mata Kuliah:

Regression Testing adalah proses pengujian perangkat lunak yang dilakukan untuk memastikan bahwa pembaruan atau perubahan dalam kode tidak mengakibatkan kesalahan pada fitur atau fungsi yang sudah ada. Ini adalah langkah penting dalam menjaga stabilitas perangkat lunak ketika ada penambahan fitur, perbaikan bug, atau perubahan lainnya. Pembelajaran ini akan membantu mahasiswa memahami defines dan konsep dasar, karakteristik, jenis-jenis, tujuan dan manfaat regression testing, serta kapan dan bagaimana menggunakannya secara efektif dalam siklus pengembangan perangkat lunak.

Tujuan Pembelajaran:

- Memahami konsep dan definisi regression testing serta peran pentingnya dalam proses pengembangan perangkat lunak.
- Mengidentifikasi jenis-jenis regression testing dan kapan masing-masing jenis perlu diterapkan.
- Mengetahui kelebihan dan kekurangan regression testing serta kapan memilih antara pengujian manual atau otomatis.
- Mempelajari alat-alat (tools) regression testing yang dapat digunakan dalam pengujian otomatis.

Materi Pembelajaran:

- Definisi Regression Testing
- Karakteristik Regression Testing
- Tujuan Regression Testing
- Jenis-jenis Regression Testing
- Manfaat Regression Testing
- Kelebihan dan Kekurangan Regression Testing
- Alat-Alat (Tools) untuk Regression Testing
- Kapan dan Bagaimana Menerapkan Regression Testing

1. Definisi Pengujian Regressi (Regression Testing)

Pengujian regressi adalah proses pengujian ulang perangkat lunak setelah adanya perubahan, pembaruan, atau modifikasi kode untuk memastikan bahwa fitur atau fungsi yang ada sebelumnya tidak terpengaruh oleh perubahan tersebut. Tujuan utama dari pengujian ini adalah untuk mendeteksi adanya bug atau kerusakan pada fungsionalitas yang sudah ada setelah sistem diperbarui. Menurut Ian Sommerville dalam bukunya *Software Engineering* (2011), pengujian regressi merupakan elemen kritis dalam memastikan stabilitas perangkat lunak setelah dilakukan modifikasi, terutama dalam konteks pengembangan yang berkelanjutan, seperti metode Agile atau DevOps.

Pengujian regressi dilakukan setiap kali ada perubahan signifikan dalam perangkat lunak, baik dalam penambahan fitur baru, perbaikan bug, atau peningkatan kinerja. Pengujian ini memastikan bahwa fungsi-fungsi yang sudah ada tetap berfungsi sebagaimana mestinya tanpa ada gangguan akibat perubahan kode baru. Misalnya, jika sebuah fitur login ditambahkan, pengujian regressi akan memverifikasi bahwa fitur lain seperti checkout, profil pengguna, dan lain-lain masih bekerja dengan benar.

Menurut penelitian oleh Runeson dan Engström (2009), pengujian regressi adalah bagian penting dari jaminan kualitas perangkat lunak yang memungkinkan tim pengembang mendeteksi dan memperbaiki bug pada tahap awal setelah perubahan dilakukan. Seiring berkembangnya perangkat lunak, pengujian regressi menjadi semakin penting karena cakupan fungsionalitas yang perlu diuji semakin besar.

Pengujian regressi juga sangat berguna dalam lingkungan pengembangan perangkat lunak yang sering mengalami perubahan atau pembaruan kode. Karena proses ini bersifat repetitif, sering kali pengujian regressi diotomatisasi dengan menggunakan alat-alat seperti Selenium, TestNG, atau JUnit untuk mengurangi waktu dan biaya yang diperlukan untuk pengujian manual yang berulang.

Secara keseluruhan, pengujian regressi memastikan stabilitas, keandalan, dan konsistensi sistem setelah perubahan, memberikan jaminan kepada tim pengembang bahwa tidak ada bug baru yang merusak fungsionalitas yang sudah ada.

2. Konsep Dasar Pengujian Regressi (Regression Testing)

Konsep dasar dari pengujian regressi adalah untuk memastikan stabilitas perangkat lunak setelah perubahan kode. Dalam siklus pengembangan modern seperti Agile dan DevOps, regresi dilakukan secara terus-menerus untuk memeriksa apakah modifikasi baru atau penambahan fitur telah menyebabkan gangguan pada fungsionalitas sebelumnya.

Framework seperti TestNG, Selenium, dan JUnit digunakan untuk melakukan pengujian regressi secara otomatis, sehingga dapat diintegrasikan ke dalam pipeline Continuous Integration/Continuous Delivery (CI/CD).

Pengujian regresi adalah proses untuk memastikan bahwa perangkat lunak tetap berfungsi dengan benar setelah ada perubahan, seperti penambahan fitur baru, perbaikan bug, atau peningkatan performa. Tujuan utamanya adalah untuk memverifikasi bahwa fitur-fitur yang sudah ada tidak terpengaruh atau rusak akibat perubahan tersebut. Setiap kali perangkat lunak mengalami modifikasi, baik kecil maupun besar, pengujian regresi dilakukan untuk menguji kembali fungsionalitas inti yang ada sebelumnya.

Pengujian ini memastikan bahwa perubahan baru tidak memperkenalkan bug atau masalah yang tidak diinginkan dalam bagian lain dari perangkat lunak. Misalnya, ketika pengembang menambahkan fitur baru di sistem login pada aplikasi, pengujian regresi akan memverifikasi bahwa fitur checkout, pembayaran, atau pengaturan akun pengguna masih berfungsi dengan benar seperti sebelumnya. Dengan kata lain, regresi adalah upaya memastikan "tidak ada kerusakan" setelah perubahan dilakukan.

Pengujian regresi sering kali membutuhkan pengujian yang berulang, dan oleh karena itu, otomatisasi regresi menjadi solusi yang efisien untuk mempercepat proses pengujian ini. Dalam proyek-proyek besar yang sering mengalami perubahan, otomatisasi regresi memungkinkan pengujian dilakukan secara terus-menerus tanpa memakan banyak waktu dan sumber daya manusia. Otomatisasi membantu memastikan bahwa perangkat lunak tetap stabil dan bebas dari bug, bahkan ketika sering diperbarui.

Dalam pengujian regresi dapat dilakukan dengan pengujian manual dan otomatisasi pengujian, Pengujian regresi manual adalah proses di mana tester menjalankan kasus uji secara manual untuk memastikan bahwa perubahan kode tidak merusak fungsionalitas yang sudah ada. Meskipun metode ini efektif untuk pengujian yang kompleks dan membutuhkan interaksi manusia, pengujian ini memakan waktu dan rentan terhadap kesalahan. Sebaliknya, pengujian regresi otomatis menggunakan alat otomatisasi seperti Selenium atau TestNG untuk menjalankan skrip pengujian berulang kali tanpa keterlibatan manual. Metode ini lebih cepat, konsisten, dan efisien, terutama untuk proyek yang sering diperbarui dan memerlukan pengujian berulang dalam skala besar. Berikut tabel perbandingan Pengujian regresi manual dan otomatis:

Aspek	Pengujian Regresi Manual	Pengujian Regresi Otomatis
Waktu dan Efisiensi	Memakan waktu lebih lama karena tester harus menguji secara manual setiap kasus uji.	Lebih cepat karena menggunakan skrip pengujian otomatis yang bisa dijalankan berulang kali.
Konsistensi dan Akurasi	Rentan terhadap kesalahan manusia, karena dilakukan secara manual.	Lebih konsisten karena pengujian otomatis tidak akan melewatkan langkah pengujian.
Skalabilitas dan Frekuensi Pengujian	Sulit diskalakan, semakin banyak kasus uji, semakin lambat proses pengujian.	Sangat dapat diskalakan, pengujian regresi dapat dilakukan berkali-kali secara otomatis.
Sumber Daya	Membutuhkan lebih banyak sumber daya manusia.	Mengurangi ketergantungan pada tester manusia setelah skrip otomatis dibuat.
Kapan Digunakan?	Digunakan ketika diperlukan verifikasi manual yang lebih mendalam atau ketika otomatisasi belum disiapkan.	Digunakan untuk pengujian yang berulang dan stabil dalam pengembangan berkelanjutan atau dalam CI/CD.

3. Karakteristik Pengujian Regressi (Regression Testing)

Pengujian regresi memiliki karakteristik utama yaitu berfokus pada pengujian berulang untuk memastikan bahwa perubahan pada kode, seperti penambahan fitur atau perbaikan bug, tidak merusak fungsionalitas yang sudah ada. Pengujian ini mencakup cakupan luas, mulai dari pengujian unit hingga sistem, dan sering diotomatisasi untuk efisiensi karena dilakukan setiap kali ada perubahan kode. Tujuannya adalah menjaga stabilitas dan kualitas perangkat lunak dengan menemukan bug baru yang mungkin muncul akibat modifikasi kode, menjadikannya penting dalam pengembangan berkelanjutan (CI/CD).

Berikut adalah penjelasan mengenai karakteristik utama pengujian regresi:

a. **Pengujian Berulang:** Pengujian regresi dilakukan berulang kali setiap kali ada perubahan atau pembaruan pada perangkat lunak. Setiap perubahan pada kode, baik besar maupun kecil, memerlukan pengujian regresi untuk memastikan bahwa fungsi-fungsi yang sudah ada sebelumnya tidak terganggu. Pengujian ini dilakukan pada fitur yang sudah ada, untuk memastikan stabilitas dan keandalan perangkat lunak.
b. **Melibatkan Pengujian Fungsionalitas yang Sudah Ada:** Salah satu karakteristik utama pengujian regresi adalah fokusnya pada fungsionalitas yang sudah ada. Pengujian ini memverifikasi bahwa fitur-fitur lama tetap berfungsi dengan baik meskipun ada fitur baru atau perubahan kode. Hal ini dilakukan dengan menjalankan kembali tes-tes lama untuk memastikan bahwa tidak ada fungsionalitas yang rusak.
c. **Mengidentifikasi Bug Baru atau Regressi:** Pengujian regresi bertujuan untuk menemukan bug baru atau regresi (penurunan kualitas) yang mungkin terjadi setelah

perubahan kode. Misalnya, jika penambahan fitur baru secara tidak sengaja merusak fitur lama, pengujian regresi akan membantu mengidentifikasi masalah ini sebelum perangkat lunak dirilis ke pengguna.

d. **Cakupan Luas:** Pengujian regresi mencakup berbagai aspek perangkat lunak. Pengujian ini bisa mencakup pengujian unit, pengujian integrasi, pengujian sistem, dan pengujian end-to-end. Semakin luas cakupan pengujian regresi, semakin baik kualitas perangkat lunak yang dihasilkan.
e. **Biasanya Diotomatisasi:** Karena sifatnya yang berulang dan sering dilakukan setiap ada perubahan kode, pengujian regresi biasanya diotomatisasi. Otomatisasi membantu menjalankan serangkaian tes secara cepat dan konsisten, sehingga mempercepat proses pengujian dan meminimalkan risiko kesalahan manusia. Alat-alat seperti Selenium, TestNG, JUnit, dan lainnya sering digunakan untuk otomatisasi pengujian regresi.
f. **Penting dalam Pengembangan Berkelanjutan (CI/CD):** Pengujian regresi sangat penting dalam pengembangan perangkat lunak berkelanjutan, terutama di lingkungan CI/CD (Continuous Integration/Continuous Deployment). Di lingkungan ini, pengujian regresi dijalankan setiap kali ada perubahan kode yang digabungkan ke dalam branch utama, sehingga menjaga kualitas perangkat lunak dalam rilis-rilis berkelanjutan.
g. **Mengutamakan Stabilitas Perangkat Lunak:** Pengujian regresi berfokus pada stabilitas perangkat lunak. Tujuannya adalah memastikan bahwa perubahan yang dilakukan tidak menyebabkan perangkat lunak menjadi tidak stabil atau menghasilkan kesalahan yang tidak diinginkan. Dengan demikian, perangkat lunak yang telah melewati pengujian regresi lebih stabil dan lebih dapat diandalkan.
h. **Terus Berkembang dengan Penambahan Kasus Uji:** Setiap kali ada fitur baru yang ditambahkan atau masalah yang diperbaiki, kasus-kasus uji baru ditambahkan ke pengujian regresi. Ini membuat suite pengujian regresi terus berkembang seiring dengan pertumbuhan perangkat lunak, sehingga cakupan pengujian menjadi lebih luas.

Contoh kasus dalam pengujian regresi Misalkan, dalam aplikasi e-commerce, ketika fitur baru seperti **"pembayaran dengan metode QR"** ditambahkan, pengujian regresi akan memverifikasi bahwa semua fitur lain seperti **login, pencarian produk, penambahan ke keranjang, dan metode pembayaran yang sudah ada tetap berfungsi dengan baik.** Pengujian regresi yang efektif sangat penting untuk menjaga kualitas perangkat lunak, terutama di proyek yang sering mengalami pembaruan. Ini membantu mengidentifikasi dan memperbaiki masalah lebih awal, mengurangi risiko bug baru yang mempengaruhi fungsionalitas lama.

4. Tujuan Pengujian Regressi (Regression Testing)

Pengujian regresi memiliki beberapa tujuan utama yang sangat penting dalam siklus pengembangan perangkat lunak. Berikut adalah tujuan-tujuan dari pengujian regresi:

a. Memastikan Stabilitas Fungsionalitas Lama.

Tujuan utama pengujian regresi adalah memastikan bahwa perubahan kode tidak mempengaruhi atau merusak fungsionalitas yang sudah ada. Fitur-fitur yang telah berfungsi dengan baik sebelumnya harus tetap berjalan dengan lancar setelah penambahan fitur baru, perbaikan bug, atau peningkatan lainnya. Ini memastikan bahwa perangkat lunak tetap stabil dan dapat diandalkan, meskipun ada modifikasi.

b. Mengidentifikasi Bug Baru yang Muncul Setelah Perubahan.

Setiap kali perubahan dilakukan pada kode perangkat lunak, ada risiko bahwa perubahan tersebut akan menyebabkan bug baru. Pengujian regresi bertujuan untuk menemukan bug atau kesalahan yang mungkin terjadi akibat modifikasi kode. Misalnya, ketika sebuah fitur baru ditambahkan, pengujian regresi akan memastikan bahwa fitur tersebut tidak secara tidak sengaja mempengaruhi fitur lain.

c. Mencegah Regressi (Penurunan Kualitas).

Regressi dalam konteks pengujian perangkat lunak mengacu pada situasi di mana perubahan atau pembaruan menyebabkan fungsionalitas yang sebelumnya baik menjadi tidak berfungsi. Pengujian regresi dilakukan untuk menghindari hal ini, sehingga perangkat lunak tetap berada pada tingkat kualitas yang tinggi meskipun telah dilakukan modifikasi.

d. Meningkatkan Keandalan Perangkat Lunak.

Dengan memastikan bahwa semua bagian perangkat lunak berfungsi dengan baik setelah setiap perubahan, pengujian regresi membantu meningkatkan keandalan perangkat lunak. Perangkat lunak yang telah melewati pengujian regresi secara konsisten lebih andal karena berbagai fungsionalitasnya telah diuji ulang dan dipastikan bekerja dengan baik setelah pembaruan.

e. Menjamin Integrasi yang Lancar.

Dalam proyek pengembangan perangkat lunak yang melibatkan banyak tim atau fitur yang saling terkait, pengujian regresi memastikan bahwa integrasi antara berbagai modul atau komponen perangkat lunak berjalan lancar. Ini sangat penting dalam proyek yang besar dan kompleks, di mana perubahan di satu bagian dapat mempengaruhi bagian lainnya.

f. Mempercepat Deteksi Masalah dalam Pengembangan Berkelanjutan (CI/CD).

Dalam lingkungan Continuous Integration/Continuous Deployment (CI/CD), pengujian regresi sangat penting untuk mendeteksi masalah secara cepat setiap kali ada perubahan kode yang digabungkan ke branch utama. Tujuannya adalah untuk memastikan bahwa perangkat lunak tetap siap dirilis dengan cepat tanpa ada penurunan kualitas atau fungsionalitas.

g. Meningkatkan Efisiensi Pengujian melalui Otomatisasi.

Pengujian regresi yang diotomatisasi mempercepat proses pengujian berulang, sehingga memastikan bahwa setiap kali perubahan dilakukan, semua fungsionalitas diuji ulang tanpa memakan banyak waktu. Ini membantu menghemat waktu dan sumber daya serta memastikan pengujian yang konsisten dan komprehensif.

h. Meningkatkan Kepuasan Pengguna.

Dengan memastikan bahwa perangkat lunak tetap berfungsi dengan baik setelah perubahan, pengujian regresi meningkatkan kepuasan pengguna. Perangkat lunak yang diuji regresi secara efektif cenderung memiliki lebih sedikit bug setelah dirilis, sehingga memberikan pengalaman pengguna yang lebih baik.

Pengujian regresi berperan penting dalam menjaga stabilitas, keandalan, dan kualitas perangkat lunak setelah perubahan, serta menjadi bagian integral dari proses pengembangan perangkat lunak modern, terutama dalam lingkungan yang sering diperbarui atau diubah.

5. Jenis-Jenis Pengujian Regressi (Regression Testing)

Pengujian regresi adalah proses pengujian perangkat lunak yang dilakukan untuk memastikan bahwa perubahan atau pembaruan tidak merusak fungsionalitas yang ada. Dalam proses pengujian regresi, terdapat beberapa jenis pengujian yang dapat digunakan, tergantung pada cakupan dan tujuan pengujian. Setiap jenis pengujian regresi memiliki fokus dan metode yang berbeda, tetapi semuanya bertujuan untuk memastikan bahwa perangkat lunak tetap berfungsi dengan baik setelah perubahan kode. Berikut adalah jenis-jenis pengujian regresi yang umum digunakan:

1. Retest All

Retest All merupakan metode di mana semua kasus uji dijalankan kembali, tanpa memandang apakah mereka terkait dengan perubahan kode atau tidak. Tujuannya adalah untuk memastikan bahwa semua bagian perangkat lunak tetap berfungsi seperti yang diharapkan. Retest All adalah proses menjalankan kembali semua kasus uji, baik yang

terkait dengan perubahan kode atau tidak, untuk memverifikasi bahwa perubahan tidak memengaruhi fungsionalitas lain. Semua kasus uji yang pernah dilakukan sebelumnya diuji ulang untuk memastikan bahwa seluruh perangkat lunak tetap berfungsi dengan benar.

Kelebihan Retest All **:**

- **Cakupan penuh:** Semua fitur dan fungsi diuji ulang sehingga tidak ada bagian yang terlewat.
- **Menjamin stabilitas keseluruhan sistem:** Dengan menguji semua bagian, ini memastikan bahwa tidak ada bagian sistem yang rusak akibat perubahan.

Kekurangan Retest All**:**

- **Sangat memakan waktu:** Karena semua kasus uji dijalankan kembali, metode ini membutuhkan waktu yang sangat lama, terutama untuk aplikasi besar.
- **Boros sumber daya:** Memerlukan sumber daya manusia dan komputasi yang besar, terutama tanpa alat otomatisasi.

2. Complete Regression Testing

Complete Regression Testing adalah proses menjalankan seluruh set kasus uji regresi setelah perubahan kode besar, seperti penambahan fitur utama atau pembaruan sistem. Complete Regression Testing dilakukan setelah perubahan kode besar untuk memastikan bahwa semua fitur dalam perangkat lunak diuji secara menyeluruh. Semua tes regresi yang ada dijalankan untuk memastikan tidak ada penurunan kualitas fungsionalitas.

Complete Regression Testing melibatkan pengujian seluruh perangkat lunak secara menyeluruh setelah setiap perubahan. Dalam jenis ini, semua kasus uji yang pernah dijalankan akan dijalankan kembali untuk memverifikasi bahwa seluruh fungsionalitas masih berfungsi dengan benar. Pengujian ini biasanya dilakukan jika perubahan yang dilakukan pada perangkat lunak sangat signifikan atau berdampak luas pada sistem. Complete Regression Testing jadi pilihan untuk digunakan ketika perubahan kode besar dilakukan, seperti penggantian arsitektur atau migrasi sistem, saat mengubah logika bisnis inti yang memengaruhi seluruh sistem.

Kelebihan Complete Regression Testing:

- **Cakupan luas:** Pengujian seluruh perangkat lunak memastikan bahwa semua fungsionalitas yang ada diuji ulang.
- **Menemukan bug yang tersembunyi:** Pengujian yang komprehensif meningkatkan kemungkinan menemukan bug yang tidak terdeteksi dalam pengujian sebelumnya.

Kekurangan Complete Regression Testing:

- **Lama dan mahal:** Sama seperti Retest All, Complete Regression Testing membutuhkan waktu dan biaya tinggi untuk pelaksanaannya.
- **Kurang efisien:** Banyak pengujian bisa saja dilakukan pada area yang tidak terpengaruh oleh perubahan, yang menyebabkan inefisiensi.

Contoh Implementasi Complete Regression Testing:

- **Studi Kasus:** Sebuah perusahaan fintech menambahkan fitur baru untuk pembayaran digital.
- **Tindakan pengujian:** Semua modul diuji, termasuk pendaftaran pengguna, transaksi pembayaran, dan manajemen akun, untuk memastikan fitur baru tidak merusak modul yang sudah ada.
- **Tujuan pengujian:** Memastikan bahwa penambahan fitur baru tidak berdampak negatif pada fungsionalitas yang sudah ada.

3. Selective Regression Testing

Selective Regression Testing adalah metode di mana hanya kasus uji yang relevan dengan modul atau bagian yang terpengaruh oleh perubahan kode yang diuji ulang. Selective Regression Testing fokus pada pengujian bagian atau modul yang secara langsung terpengaruh oleh perubahan kode. Hanya kasus uji yang relevan dengan area yang terpengaruh yang diuji ulang, membuat proses ini lebih efisien daripada Retest All atau Complete Regression Testing.

Pengujian regresi selektif melibatkan pemilihan sebagian dari semua kasus uji yang dijalankan kembali berdasarkan perubahan spesifik yang dilakukan pada kode. Hanya kasus uji yang terkait langsung dengan perubahan atau area yang terdampak oleh modifikasi yang akan dijalankan. Jenis ini lebih efisien dalam hal waktu dan sumber daya karena hanya pengujian yang relevan yang dilakukan. Selective Regression Testing digunakan dalam situasi berikut:

- Ketika perubahan kode hanya berdampak pada bagian-bagian tertentu dari sistem.
- Dalam pengembangan berkelanjutan, saat fitur kecil atau perbaikan bug dilakukan.

Kelebihan Selective Regression Testing**:**

- **Efisien waktu dan sumber daya:** Pengujian hanya dilakukan pada bagian-bagian yang terpengaruh, sehingga menghemat waktu dan sumber daya.

- **Cakupan terfokus:** Menyasar bagian spesifik yang kemungkinan besar terpengaruh oleh perubahan.

Kekurangan Selective Regression Testing**:**

- **Risiko melewatkan bug:** Ada kemungkinan bug tersembunyi pada bagian perangkat lunak yang tidak diuji karena tidak teridentifikasi terpengaruh oleh perubahan.
- **Membutuhkan pemahaman mendalam:** Pengembang dan penguji harus memiliki pemahaman yang mendalam tentang sistem untuk mengidentifikasi bagian yang terpengaruh.

Contoh Implementasi Selective Regression Testing:

- **Studi Kasus:** Sebuah tim pengembang memperbaiki bug kecil di modul pembayaran pada aplikasi e-commerce.
- **Tindakan pengujian:** Hanya kasus uji terkait pembayaran, pemrosesan kartu kredit, dan penanganan kegagalan pembayaran yang dijalankan.
- **Tujuan pengujian:** Menghemat waktu dan sumber daya dengan hanya memfokuskan pada bagian yang t

4. Partial Regression Testing

Partial Regression Testing dilakukan untuk memastikan bahwa modul atau bagian dari perangkat lunak yang baru saja dimodifikasi dapat bekerja dengan baik, tanpa memengaruhi fungsionalitas lain yang berinteraksi dengannya. Pengujian difokuskan pada bagian yang diperbaiki serta modul-modul yang terkait. Pengujian regresi sebagian dilakukan setelah perubahan dilakukan pada modul tertentu, dan tujuan utamanya adalah untuk memastikan bahwa perubahan tersebut tidak memengaruhi bagian lain dari perangkat lunak yang berfungsi baik. Pengujian ini fokus pada modul yang baru saja dimodifikasi serta area yang secara langsung terhubung dengan modul tersebut. Partial Regression Testing digunakan Ketika perubahan kode dilakukan pada satu modul atau komponen sistem tanpa memengaruhi keseluruhan perangkat lunak.

Kelebihan Partial Regression Testing**:**

- **Efisien:** Pengujian hanya dilakukan pada area yang terkait dengan perubahan, menghemat waktu dan sumber daya.
- **Mengurangi risiko bug interaksi:** Mengidentifikasi masalah pada bagian sistem yang berinteraksi dengan bagian yang dimodifikasi.

Kekurangan Partial Regression Testing:

- **Risiko kehilangan cakupan:** Ada kemungkinan bagian lain dari sistem yang tidak terkait langsung dengan perubahan terlewat dari pengujian.
- **Bergantung pada dokumentasi yang baik:** Memerlukan pemahaman yang jelas tentang dependensi antar modul.

Contoh Implementasi Partial Regression Testing:

- **Studi Kasus:** Modul pengiriman pada sistem manajemen gudang diperbarui untuk mendukung metode pengiriman baru.
- **Tindakan pengujian:** Selain pengujian pada modul pengiriman, dilakukan juga pengujian pada modul manajemen inventaris dan pelacakan pesanan untuk memastikan perubahan tidak memengaruhi aliran data antara modul-modul tersebut.
- **Tujuan pengujian:** Memastikan bahwa pembaruan modul tidak menyebabkan masalah pada bagian sistem yang terkait.

5. Progressive Regression Testing

Progressive Regression Testing dilakukan saat ada perubahan besar atau pembaruan pada perangkat lunak yang sedang dikembangkan. Tujuan pengujian ini adalah untuk memverifikasi bahwa fitur baru yang ditambahkan tidak merusak fungsionalitas yang sudah ada, serta memastikan bahwa fitur baru bekerja dengan baik dalam konteks sistem secara keseluruhan.

Pengujian regresi progresif dilakukan ketika perubahan atau pembaruan dilakukan pada perangkat lunak yang sedang dalam tahap pengembangan. Pengujian ini bertujuan untuk memastikan bahwa perubahan baru tidak mengganggu fitur yang sedang dalam pengembangan atau fitur yang sudah diuji sebelumnya. Progressive Regression Testing digunakan dalam lingkungan pengembangan berkelanjutan atau agile, di mana pembaruan sering dilakukan.

Kelebihan Progressive Regression Testing:

- **Efektif dalam pengembangan berkelanjutan:** Cocok untuk metodologi pengembangan Agile di mana fitur-fitur baru sering ditambahkan.
- **Memastikan kompatibilitas fitur baru:** Fitur baru diuji bersama dengan fungsionalitas lama untuk memastikan mereka bekerja dengan baik secara bersamaan.

Kekurangan Progressive Regression Testing:

- **Membutuhkan waktu dalam implementasi:** Setiap kali ada perubahan baru, pengujian perlu dilakukan secara progresif.
- **Dapat menambah kompleksitas:** Memerlukan manajemen yang baik untuk memisahkan pengujian untuk fitur baru dan pengujian regresi.

Contoh Implementasi Progressive Regression Testing:

- **Studi Kasus:** Sebuah aplikasi manajemen proyek menambahkan fitur pelaporan baru.
- **Tindakan pengujian:** Fitur pelaporan baru diuji bersamaan dengan modul manajemen proyek, modul tugas, dan modul pelacakan waktu untuk memastikan kompatibilitas.
- **Tujuan pengujian:** Memastikan bahwa fitur baru berfungsi baik dengan sistem yang sudah ada dan tidak menimbulkan masalah pada fungsionalitas lain.

6. Automated Regression Testing

Automated Regression Testing melibatkan penggunaan alat otomatisasi untuk menjalankan kembali pengujian regresi, khususnya pengujian yang sering diulang setelah setiap perubahan kode. Hal ini sangat membantu dalam pengembangan perangkat lunak yang berkelanjutan. Automated Regression Testing menggunakan alat otomatisasi untuk menjalankan pengujian regresi berulang. Alat otomatisasi digunakan untuk menjalankan kembali kasus uji setiap kali ada perubahan kode baru, membuat proses lebih cepat dan lebih efisien dibandingkan pengujian manual. Karena pengujian regresi sering kali melibatkan pengulangan pengujian yang sama setiap kali perubahan kode dilakukan, otomatisasi membantu mempercepat proses dan meningkatkan konsistensi hasil pengujian. Alat otomatisasi seperti Selenium, JUnit, dan TestNG dapat digunakan untuk menjalankan pengujian regresi secara otomatis. Automated Regression Testing digunakan dalam untuk pengujian berikut:

- Dalam proyek besar dengan siklus pengembangan berkelanjutan.
- Ketika pengujian regresi perlu dilakukan secara berkala setelah setiap perubahan kode.

Kelebihan Automated Regression:

- **Cepat dan efisien:** Menjalankan pengujian berulang dengan alat otomatisasi menghemat waktu dan tenaga.
- **Konsisten:** Mengurangi risiko kesalahan manusia dalam pengujian berulang.

Kekurangan Automated Regression:

- **Biaya awal tinggi:** Investasi awal dalam alat otomatisasi dan pembuatan skrip pengujian bisa mahal.
- **Pemeliharaan:** Skrip otomatis perlu diperbarui sesuai dengan perubahan perangkat lunak.

Contoh Implementasi Automated Regression:

- **Studi Kasus:** Sebuah aplikasi SaaS (Software as a Service) menjalankan pengujian otomatis menggunakan Selenium setiap kali fitur baru ditambahkan ke dalam platform.
- **Tindakan pengujian:** Pengujian regresi otomatis dilakukan pada semua modul, termasuk login, manajemen akun, dan pembayaran, untuk memastikan tidak ada masalah yang muncul setelah penambahan fitur.
- **Tujuan pengujian:** Mengurangi waktu yang diperlukan untuk pengujian manual dan memastikan bahwa perangkat lunak diuji secara konsisten dengan setiap perubahan.

7. Priority-based Regression Testing

Priority-based Regression Testing adalah pendekatan dalam pengujian regresi yang memfokuskan pengujian pada kasus-kasus atau area fungsional tertentu dalam perangkat lunak berdasarkan prioritas. Pendekatan ini dilakukan untuk mengoptimalkan waktu dan sumber daya dengan memberikan perhatian khusus pada fitur atau fungsi perangkat lunak yang dianggap paling penting atau paling rentan terhadap perubahan. Priority-based Regression Testing biasanya diterapkan dalam proyek besar atau ketika waktu pengujian terbatas, sehingga memungkinkan pengembang untuk memfokuskan pengujian pada bagian-bagian kritis perangkat lunak yang kemungkinan besar terdampak oleh perubahan terbaru. Priority-based Regression Testing digunakan ketika:

- **Waktu Pengujian Terbatas:** Misalnya, saat menghadapi tenggat waktu yang ketat dan tidak memungkinkan untuk melakukan pengujian menyeluruh.
- **Fitur atau Fungsi Utama Berubah:** Ketika perubahan besar terjadi pada komponen inti aplikasi, yang berisiko mempengaruhi bagian-bagian lain.
- **Sumber Daya Terbatas:** Ketika ada keterbatasan dalam tim pengujian, alat, atau infrastruktur untuk melakukan pengujian yang komprehensif.

- **Frekuensi Rilis yang Tinggi:** Dalam pengembangan perangkat lunak Agile atau DevOps, di mana ada rilis berkelanjutan, pengujian regresi berbasis prioritas memungkinkan pengujian cepat pada area utama sebelum rilis.

Kelebihan Priority-based Regression Testing

- **Efisiensi Waktu:** Mengurangi waktu pengujian dengan berfokus pada bagian yang paling kritis atau rawan kesalahan.
- **Optimalisasi Sumber Daya:** Mencapai cakupan pengujian yang lebih baik dengan sumber daya terbatas, terutama di bawah batasan waktu atau tenaga kerja.
- **Deteksi Dini Masalah:** Fokus pada area berisiko tinggi meningkatkan kemungkinan menemukan masalah atau regresi kritis lebih awal.
- **Cocok untuk Proyek Besar:** Mengurangi beban pengujian pada proyek besar dengan mengutamakan bagian yang rentan atau penting saja.

Kekurangan Priority-based Regression Testing

- Tidak Komprehensif: Karena hanya area prioritas yang diuji, ada risiko bagian lain terlewatkan yang mungkin juga terdampak oleh perubahan.
- Pengabaian Fitur Lain: Fitur atau fungsi yang tidak diprioritaskan mungkin tidak diuji, sehingga mengabaikan masalah potensial yang bisa mempengaruhi pengguna.
- Kesulitan Menetapkan Prioritas: Penentuan prioritas yang tidak akurat atau subjektif dapat mengarah pada hasil pengujian yang tidak representatif atau relevan.
- Ketergantungan pada Analisis Dampak: Membutuhkan analisis dampak yang kuat untuk mengidentifikasi bagian-bagian prioritas yang harus diuji, yang bisa memakan waktu.

Contoh Implementasi Priority-based Regression Testing:

- **Studi Kasus:** Sebuah aplikasi e-commerce mengalami perubahan pada modul pembayaran dengan penambahan metode pembayaran baru.
- **Tindakan pengujian:** Identifikasi Area Prioritas Tinggi, Persiapan Kasus Uji, Eksekusi Pengujian, Dokumentasi dan Perbaikan.
- **Tujuan pengujian:** Memastikan fitur utama seperti pembayaran dan checkout berfungsi baik pasca-perubahan, sambil menghemat waktu dengan fokus pada area prioritas tinggi.

Berbagai jenis pengujian regresi, mulai dari Retest All hingga Progressive Regression Testing, memberikan fleksibilitas dalam memilih cakupan pengujian berdasarkan tingkat perubahan pada perangkat lunak. Implementasi praktis dari jenis-jenis pengujian ini tergantung pada sifat dan skala perubahan kode. Automasi juga dapat mempercepat pengujian regresi, terutama dalam skenario pengembangan berkelanjutan di mana pengujian berulang sangat dibutuhkan.

6. Manfaat Pengujian Regressi (Regression Testing)

Pengujian regresi merupakan bagian penting dari siklus pengembangan perangkat lunak yang memastikan bahwa perubahan, perbaikan bug, atau penambahan fitur baru tidak mengganggu fungsionalitas yang sudah ada. Berikut adalah beberapa manfaat utama dari pengujian regresi:

1. **Menjamin Stabilitas Perangkat Lunak.**
   Pengujian regresi memastikan bahwa perubahan yang dilakukan pada perangkat lunak, baik itu perbaikan bug, penambahan fitur, atau peningkatan kinerja, tidak menyebabkan kerusakan atau gangguan pada fitur yang sudah berfungsi. Pengujian ini memverifikasi bahwa bagian perangkat lunak yang tidak terpengaruh langsung oleh perubahan masih berfungsi sesuai harapan.
2. **Mengurangi Risiko Kegagalan di Produksi.**
   Melalui pengujian regresi, bug atau kesalahan yang tidak terdeteksi saat perubahan dilakukan dapat ditemukan sebelum perangkat lunak dirilis ke lingkungan produksi. Ini secara signifikan mengurangi risiko kegagalan sistem atau kerusakan fungsionalitas penting setelah peluncuran perangkat lunak.
3. **Mendukung Pengembangan Berkelanjutan (Continuous Development).**
   Dalam lingkungan pengembangan modern, seperti Agile dan DevOps, pembaruan perangkat lunak terjadi secara iteratif dan berkelanjutan. Pengujian regresi membantu memastikan bahwa setiap perubahan yang terjadi dalam siklus pengembangan tidak mengganggu stabilitas sistem secara keseluruhan. Otomatisasi pengujian regresi dapat mengurangi beban pengujian berulang setiap kali ada pembaruan.
4. **Mengidentifikasi Bug Baru dengan Cepat.**
   Ketika fitur atau komponen baru ditambahkan ke sistem, seringkali bug baru bisa muncul karena interaksi yang tidak terduga antara komponen baru dan yang sudah ada. Pengujian regresi membantu mendeteksi bug baru yang mungkin tidak terlihat selama pengujian fungsional atau pengujian unit.

5. **Meningkatkan Efisiensi dengan Otomatisasi.**

   Pengujian regresi yang diotomatisasi memungkinkan pengujian dilakukan dengan lebih cepat dan berulang tanpa memerlukan campur tangan manual. Ini sangat mengurangi waktu pengujian, terutama pada sistem yang sering mengalami perubahan. Selain itu, otomatisasi memungkinkan pengujian regresi dilakukan secara konsisten dan efisien, bahkan untuk perangkat lunak yang sangat kompleks.

6. **Meningkatkan Kepercayaan terhadap Perangkat Lunak.**

   Dengan melakukan pengujian regresi yang komprehensif, tim pengembang, penguji, dan pemangku kepentingan lainnya mendapatkan keyakinan bahwa perangkat lunak tidak hanya berfungsi sesuai spesifikasi, tetapi juga stabil dan andal meskipun ada perubahan atau peningkatan. Hal ini dapat meningkatkan kepercayaan pelanggan terhadap kualitas perangkat lunak, sehingga meminimalkan risiko terjadinya keluhan setelah peluncuran.

Pengujian regresi memberikan banyak manfaat yang signifikan dalam siklus hidup perangkat lunak, termasuk menjamin stabilitas fungsionalitas yang ada, mengurangi risiko bug di lingkungan produksi, dan mendukung pengembangan berkelanjutan. Dengan menggunakan alat otomatisasi pengujian, pengujian regresi dapat dilakukan lebih efisien dan efektif, yang pada akhirnya membantu menjaga kualitas perangkat lunak secara keseluruhan serta meningkatkan kepercayaan pengguna terhadap perangkat lunak tersebut.

7. Kelebihan dan Kekurangan Pengujian Regressi (Regression Testing)

Pengujian regresi (Regression Testing) merupakan langkah kritis dalam siklus pengembangan perangkat lunak, terutama ketika perubahan dilakukan pada aplikasi, seperti penambahan fitur baru, perbaikan bug, atau peningkatan kinerja. Berikut adalah kelebihan dan kekurangan dari pengujian regresi.

- **Kelebihan Pengujian Regresi:**
  1. **Menjaga Stabilitas Sistem:** Pengujian regresi secara konsisten memastikan bahwa fungsionalitas yang sudah ada dalam perangkat lunak tetap stabil setelah terjadi perubahan. Hal ini penting karena fitur baru, modifikasi, atau bug fix dapat berdampak pada bagian perangkat lunak yang tidak diubah secara langsung.
  2. **Mendeteksi Bug Baru Lebih Awal:** Dengan melakukan pengujian regresi, bug atau kesalahan baru yang tidak terdeteksi dalam pengujian sebelumnya dapat ditemukan lebih awal sebelum perangkat lunak dirilis ke lingkungan produksi. Hal ini mengurangi risiko bug yang menyebabkan kerusakan besar.

3. **Mendukung Pengembangan Berkelanjutan (Continuous Development):** Dalam metodologi Agile atau DevOps, di mana pembaruan perangkat lunak terjadi secara berkelanjutan, pengujian regresi yang otomatis sangat membantu. Pengujian regresi mendukung rilis yang lebih cepat dengan memastikan bahwa fitur baru tidak merusak fungsionalitas yang ada.
4. **Meningkatkan Kepercayaan terhadap Perangkat Lunak:** Pengujian regresi memastikan bahwa perangkat lunak berfungsi secara konsisten bahkan setelah perubahan. Ini memberi keyakinan pada pengembang, penguji, dan pengguna bahwa perangkat lunak tetap stabil, yang meningkatkan kepercayaan terhadap kualitas perangkat lunak.
5. **Efisiensi Melalui Otomatisasi:** Pengujian regresi sangat cocok untuk diotomatisasi karena tes yang sama sering kali dijalankan berulang kali. Dengan menggunakan alat otomatisasi, pengujian dapat dilakukan secara lebih cepat dan efisien, menghemat waktu dan tenaga untuk pengujian manual yang berulang.

- **Kekurangan Pengujian Regresi:**
  1. **Memerlukan Waktu dan Sumber Daya yang Signifikan:** Pengujian regresi, terutama yang dilakukan secara manual, memakan waktu dan sumber daya yang signifikan. Semakin besar aplikasi dan semakin banyak fitur yang diuji, semakin lama waktu yang dibutuhkan untuk menjalankan pengujian regresi.
  2. **Biaya Implementasi Alat Otomatisasi:** Meskipun otomatisasi pengujian regresi dapat meningkatkan efisiensi, biaya implementasi alat otomatisasi dan pemeliharaannya cukup tinggi. Tim perlu melatih staf, membeli alat, dan membangun infrastruktur untuk menjalankan pengujian regresi secara otomatis.
  3. **Tidak Selalu Mendukung Pengujian Non-Fungsional:** Pengujian regresi lebih fokus pada fungsionalitas dan stabilitas perangkat lunak, sehingga kadang-kadang pengujian ini tidak mendeteksi masalah non-fungsional, seperti kinerja sistem, keamanan, atau skalabilitas, yang juga sangat penting bagi kesuksesan perangkat lunak.
  4. **Pemeliharaan Skrip Pengujian yang Kompleks:** Dalam pengujian regresi otomatis, skrip pengujian harus terus diperbarui dan dipelihara seiring perubahan pada perangkat lunak. Ini dapat menambah kompleksitas pengujian, terutama jika perangkat lunak mengalami perubahan yang signifikan atau fitur-fitur baru terus ditambahkan.

5. **Overhead pada Pengujian Manual:** Jika pengujian regresi dilakukan secara manual, beban penguji meningkat karena mereka harus mengulang kembali pengujian terhadap fungsionalitas yang sudah ada setiap kali ada pembaruan. Hal ini tidak efisien, memakan waktu, dan bisa menyebabkan kelelahan pada tim QA.

Pengujian regresi menawarkan banyak keuntungan, terutama dalam memastikan stabilitas perangkat lunak dan mendukung pengembangan berkelanjutan. Namun, tantangan dalam hal biaya, waktu, dan pemeliharaan alat otomatisasi membuatnya perlu direncanakan dengan baik. Pengujian regresi sangat cocok untuk diotomatisasi agar lebih efisien, terutama dalam proyek-proyek besar dengan banyak fitur yang saling terkait.

8. Alat-Alat (Tools) Pengujian Regressi (Regression Testing)

Alat pengujian regresi (Regression Testing Tools) digunakan untuk mengotomatiskan proses pengujian regresi, sehingga pengujian dapat dilakukan lebih efisien, konsisten, dan dalam skala yang lebih besar. Penggunaan alat-alat ini membantu mengurangi beban pengujian manual yang berulang, terutama ketika perangkat lunak mengalami banyak perubahan atau pembaruan. Berikut adalah beberapa alat (tools) pengujian regresi yang populer dan banyak digunakan di industri perangkat lunak, lengkap dengan kelebihan, fitur utama, dan kasus penggunaannya.

1. **Selenium**

   Selenium adalah salah satu alat otomatisasi pengujian yang paling populer dan digunakan secara luas, terutama untuk pengujian regresi pada aplikasi web. Selenium mendukung berbagai bahasa pemrograman seperti Java, C#, Python, dan JavaScript, serta dapat digunakan untuk menguji aplikasi di berbagai browser dan sistem operasi.

   Fitur Utama Selenium:

   - Otomatisasi pengujian di berbagai browser (Chrome, Firefox, Edge, dll.)
   - Mendukung berbagai bahasa pemrograman (Java, Python, C#, Ruby)
   - Dapat digunakan dengan berbagai alat CI/CD (Continuous Integration/Continuous Deployment)
   - Mendukung pengujian regresi yang terintegrasi dengan skrip otomatis

   Kelebihan Selenium:

   - Open source (gratis) dan banyak komunitas pengguna yang mendukung
   - Fleksibel dan dapat dikustomisasi untuk berbagai jenis pengujian regresi

- Integrasi yang baik dengan framework pengujian lainnya seperti TestNG, JUnit, dan Jenkins.

Kekurangan Selenium:

- Memerlukan keahlian pemrograman untuk menggunakannya secara efektif
- Pemeliharaan skrip otomatis memerlukan waktu, terutama jika aplikasi sering berubah.

2. **TestNG.**

TestNG adalah framework pengujian yang banyak digunakan untuk mengelola pengujian unit, pengujian integrasi, dan pengujian regresi dalam proyek perangkat lunak berbasis Java. TestNG mendukung pengujian yang terorganisir dan efisien dengan fitur eksekusi paralel dan konfigurasi yang fleksibel.

Fitur Utama TestNG:

- Pengelompokan pengujian
- Dukungan untuk eksekusi pengujian paralel
- Anotasi yang mudah digunakan untuk mendefinisikan pengujian
- Integrasi dengan Selenium untuk pengujian regresi otomatis

Kelebihan TestNG:

- Memungkinkan eksekusi pengujian otomatis yang lebih cepat dengan dukungan untuk pengujian paralel
- Mendukung pelaporan hasil pengujian yang terintegrasi dengan alat CI/CD seperti Jenkins
- Sangat cocok untuk pengujian regresi yang kompleks pada aplikasi Java

Kekurangan TestNG:

- Terutama digunakan untuk aplikasi berbasis Java
- Membutuhkan pengetahuan pemrograman untuk menulis dan memelihara skrip pengujian.

3. **Jenkins**

Jenkins adalah alat integrasi berkelanjutan (Continuous Integration) yang memungkinkan otomatisasi pengujian regresi selama pengembangan perangkat lunak. Dengan Jenkins, pengujian regresi dapat dijalankan secara otomatis setiap kali ada perubahan kode, sehingga memastikan bahwa perubahan tidak mempengaruhi fungsionalitas yang ada.

Fitur Utama Jenkins:

- Otomatisasi tugas pengujian regresi dan integrasi terus menerus

- Dukungan untuk plugin pengujian regresi, seperti Selenium dan TestNG
- Dukungan pipeline pengembangan perangkat lunak untuk pengujian berulang

Kelebihan Jenkins:

- Dapat digunakan untuk mengotomatiskan seluruh pipeline pengembangan, mulai dari build hingga pengujian regresi
- Memungkinkan pengujian otomatis di berbagai lingkungan secara paralel
- Mendukung berbagai jenis alat pengujian regresi melalui plugin

Kekurangan Jenkins:

- Membutuhkan konfigurasi yang lebih kompleks untuk integrasi penuh dengan alat pengujian
- Pemeliharaan pipeline otomatis bisa menjadi rumit jika aplikasi memiliki banyak komponen.

4. **Katalon Studio**

Katalon Studio adalah platform pengujian otomatis yang user-friendly dan mendukung pengujian regresi pada berbagai jenis aplikasi, termasuk web, mobile, dan API. Katalon Studio menawarkan antarmuka pengguna yang mudah digunakan dan berbagai alat built-in untuk otomatisasi pengujian regresi, tanpa perlu keahlian pemrograman tingkat lanjut.

Fitur Utama Katalon Studio:

- Otomatisasi pengujian regresi untuk aplikasi web, mobile, dan API
- Integrasi dengan alat CI/CD seperti Jenkins, Git, dan Docker
- Dukungan untuk skrip pengujian codeless (tanpa coding) dan berbasis kode (coding)

Kelebihan Katalon Studio:

- Mudah digunakan oleh penguji yang bukan pemrogram
- Mendukung pengujian regresi di berbagai platform (web, mobile, API)
- Integrasi yang kuat dengan alat DevOps

Kekurangan Katalon Studio:

- Versi gratis memiliki fitur terbatas dibandingkan dengan versi berbayar
- Kurang fleksibel dibandingkan alat berbasis kode seperti Selenium untuk kebutuhan pengujian yang sangat khusus.

5. **Appium**

Appium adalah alat pengujian otomatis yang dirancang untuk pengujian regresi pada aplikasi mobile, baik Android maupun iOS. Alat ini mendukung otomatisasi pengujian

regresi melalui berbagai framework seperti Selenium, sehingga memungkinkan pengujian regresi dilakukan lintas platform.

Fitur Utama Appium:

- Otomatisasi pengujian regresi untuk aplikasi Android dan iOS
- Dukungan untuk berbagai framework pengujian, termasuk Selenium WebDriver
- Mendukung pengujian lintas platform menggunakan API WebDriver yang sama

Kelebihan Appium:

- Alat gratis dan open-source
- Mendukung pengujian regresi untuk berbagai aplikasi mobile dengan satu alat
- Dapat diintegrasikan dengan CI/CD untuk pengujian regresi berulang

Kekurangan Appium:

- Memerlukan waktu setup yang cukup lama
- Kompleksitas dalam pengujian fitur-fitur khusus perangkat mobile, seperti gestur dan sensor.

6. Junit

JUnit adalah framework pengujian yang banyak digunakan untuk aplikasi Java. Meskipun biasanya digunakan untuk pengujian unit, JUnit juga dapat digunakan dalam pengujian regresi dengan bantuan alat integrasi berkelanjutan. JUnit mendukung pengujian otomatis dengan anotasi dan eksekusi pengujian paralel.

Fitur Utama JUnit:

- Mendukung anotasi pengujian untuk manajemen yang mudah
- Dukungan untuk eksekusi pengujian secara otomatis
- Integrasi yang baik dengan alat CI/CD seperti Jenkins

Kelebihan JUnit:

- Framework pengujian Java yang sederhana dan banyak digunakan
- Dapat digunakan untuk mengotomatisasi pengujian regresi pada aplikasi Java
- Mudah diintegrasikan dengan berbagai alat lain untuk otomatisasi pengujian regresi

Kekurangan JUnit:

- Terbatas pada aplikasi berbasis Java
- Tidak sefleksibel framework yang lebih modern seperti TestNG.

7. Cypress

Cypress adalah alat otomatisasi pengujian berbasis JavaScript yang banyak digunakan untuk pengujian aplikasi web, termasuk regression testing. Regression testing dengan

Cypress menjadi lebih efisien dan efektif karena fitur-fiturnya yang kuat untuk mengidentifikasi perubahan atau regresi pada aplikasi setelah pembaruan kode.

Fitur Utama Cypress:

- Real-Time Reloading: Cypress menyediakan fitur reloading secara langsung sehingga setiap perubahan dalam kode pengujian atau aplikasi akan segera terlihat. Ini membantu dalam mendeteksi kesalahan dengan cepat selama regression testing.
- Time Travel: Cypress menyimpan snapshot dari setiap langkah pengujian, memungkinkan tester untuk "mengunjungi kembali" setiap langkah dan memverifikasi bagaimana aplikasi berperilaku pada setiap titik dalam pengujian. Ini sangat membantu dalam menganalisis regresi yang kompleks.
- Automatic Waiting: Cypress otomatis menunggu untuk elemen halaman web yang dimuat dan interaksi pengguna selesai sebelum melanjutkan ke langkah berikutnya. Ini memastikan bahwa tidak ada ketergantungan pada waktu tunggu tambahan dalam pengujian regresi, meningkatkan stabilitas pengujian.
- Easy Integration with CI/CD Pipelines: Cypress dapat diintegrasikan dengan pipeline CI/CD, seperti Jenkins, CircleCI, atau GitLab, yang memungkinkan regression testing dijalankan secara otomatis setiap kali ada perubahan kode, memberikan deteksi dini terhadap regresi.
- Comprehensive API Testing Capabilities: Cypress juga mendukung pengujian API yang memungkinkan pengujian regresi end-to-end pada aplikasi web yang memiliki banyak titik integrasi API. Hal ini memastikan bahwa setiap perubahan dalam API tidak menyebabkan regresi di bagian lain dari aplikasi.

Kelebihan Cypress:

- Kecepatan dan Efisiensi: Cypress menggunakan teknologi DOM langsung yang mempercepat interaksi dengan halaman web, membuat pengujian regresi lebih cepat dibandingkan dengan banyak alat pengujian lainnya.
- Dukungan untuk JavaScript/TypeScript: Bagi tim pengembang yang sudah familiar dengan JavaScript atau TypeScript, Cypress menjadi pilihan ideal untuk regression testing karena pengujian ditulis dalam bahasa yang sudah dikenal.
- Integrasi Mudah dengan Framework Modern: Cypress bekerja baik dengan framework modern seperti React, Vue, dan Angular, yang banyak digunakan dalam aplikasi web modern, memudahkan regression testing pada aplikasi berbasis framework tersebut.

- Desain UI yang User-Friendly: Cypress menyediakan dasbor interaktif untuk melihat status setiap pengujian, memungkinkan tim QA dan pengembang memahami hasil regression testing secara visual dan lebih cepat mengidentifikasi masalah.

Kekurangan Cypress:

- Tidak Mendukung Multi-Tab atau Multi-Browser Testing: Cypress saat ini hanya mendukung pengujian di satu tab dan di Chrome serta Edge. Ini menjadi keterbatasan ketika regression testing diperlukan pada banyak browser atau skenario multi-tab.
- Tantangan dalam Pengujian File Upload dan Unduh: Meski mendukung pengujian banyak aspek aplikasi web, fitur pengujian unggah dan unduh file di Cypress masih membutuhkan beberapa solusi kreatif dan mungkin tidak sesederhana alat pengujian lainnya.
- Tidak Ideal untuk Aplikasi Non-JavaScript: Cypress berfokus pada aplikasi berbasis JavaScript. Ini berarti bahwa regression testing untuk aplikasi berbasis teknologi selain JavaScript mungkin tidak optimal jika dilakukan dengan Cypress.
- Keterbatasan Pengujian Latar Belakang (Headless Testing): Meskipun mendukung pengujian latar belakang, Cypress membutuhkan konfigurasi tambahan dan mungkin tidak bekerja sebaik alat lain di lingkungan pengujian tanpa antarmuka pengguna.

Pengujian regresi sangat penting untuk memastikan stabilitas perangkat lunak setelah ada perubahan atau penambahan fitur. Berbagai alat otomatisasi, seperti Selenium, TestNG, dan Jenkins, memberikan dukungan yang kuat untuk melakukan pengujian regresi yang konsisten dan efisien. Pilihan alat bergantung pada jenis aplikasi yang diuji, lingkungan pengembangan, dan kebutuhan pengujian spesifik. Dengan menggunakan alat otomatisasi, pengujian regresi dapat dilakukan lebih cepat dan lebih efektif, yang pada akhirnya meningkatkan kualitas perangkat lunak secara keseluruhan.

9. Kapan dan Bagaimana Menerapkan Regression Testing.

Pengujian regresi (Regression Testing) adalah proses pengujian yang dilakukan setelah adanya perubahan atau penambahan pada kode perangkat lunak untuk memastikan bahwa perubahan tersebut tidak mempengaruhi fungsionalitas yang ada. Pengujian regresi memastikan bahwa perangkat lunak tetap berfungsi seperti yang diharapkan setelah adanya

modifikasi kode, baik untuk memperbaiki bug, menambah fitur baru, atau melakukan peningkatan sistem.

Kapan Menerapkan Pengujian Regresi seperti yang sudah dijelaskan sedikit pada jenis-jenis pengujian regresi dimana penerapannya tergantung adari situasi dan kondisi atau kasus dalam pengujian perangkat lunak. Pengujian regresi dilakukan dalam beberapa situasi penting dalam siklus pengembangan perangkat lunak.

Berikut ini adalah penerapan pengujian regresi:

1. **Setelah Perbaikan Bug:** Ketika ada bug atau cacat yang diperbaiki dalam sistem, pengujian regresi diperlukan untuk memastikan bahwa perbaikan tersebut tidak memperkenalkan bug baru di bagian lain dari perangkat lunak. Bug fix bisa berdampak pada modul atau komponen lain yang berhubungan, sehingga pengujian regresi memastikan bahwa semua bagian perangkat lunak tetap stabil setelah perbaikan.
2. **Setelah Penambahan Fitur Baru:** Setiap kali fitur baru ditambahkan, pengujian regresi dilakukan untuk memastikan bahwa fitur baru tersebut tidak mengganggu atau merusak fungsionalitas yang sudah ada. Fitur baru mungkin mengubah logika bisnis atau komponen inti aplikasi, sehingga pengujian regresi diperlukan untuk menghindari masalah baru.
3. **Setelah Melakukan Peningkatan Sistem:** Pengujian regresi harus dilakukan setelah adanya peningkatan atau refactoring kode, pengoptimalan, atau perubahan pada infrastruktur (misalnya upgrade database, peningkatan platform). Hal ini untuk memastikan bahwa peningkatan tersebut tidak menyebabkan kegagalan atau masalah pada fungsionalitas yang telah ada.
4. **Pengujian Berulang di Setiap Iterasi Pengembangan Agile:** Dalam pengembangan berbasis Agile, perubahan kode atau fitur sering terjadi di setiap iterasi (sprint). Pengujian regresi dilakukan secara berkala untuk memastikan bahwa fungsionalitas yang sudah dikembangkan pada sprint sebelumnya tetap berjalan dengan baik setelah ada penambahan atau modifikasi kode pada sprint berikutnya.

Bagaimana Menerapkan Pengujian Regresi:

1. **Identifikasi Area yang Terpengaruh:** Langkah pertama dalam menerapkan pengujian regresi adalah mengidentifikasi area atau modul dalam perangkat lunak yang terpengaruh oleh perubahan kode. Pengujian tidak selalu harus mencakup seluruh aplikasi, melainkan cukup fokus pada area yang rentan atau terkait dengan perubahan.

2. **Seleksi dan Pembuatan Test Case:** Test case untuk pengujian regresi biasanya terdiri dari kasus pengujian yang sudah ada, yang mencakup fungsionalitas penting dari sistem. Test case ini harus dipilih dan diperbarui sesuai dengan area yang mengalami perubahan. Automasi pengujian regresi memungkinkan test case tersebut dijalankan secara cepat dan konsisten setiap kali ada pembaruan.
3. **Automasi Pengujian Regresi:** Karena pengujian regresi bersifat berulang, penggunaan alat otomasi sangat direkomendasikan untuk mengurangi beban manual. Automasi membantu memastikan bahwa pengujian dilakukan dengan cepat dan konsisten. Tool seperti Selenium, TestNG, atau Katalon Studio digunakan untuk menjalankan test case otomatis setiap kali ada perubahan kode yang didorong ke pipeline CI/CD (Continuous Integration/Continuous Delivery).
4. **Jalankan Pengujian Regresi Secara Paralel:** Pengujian regresi dapat dijalankan secara paralel dengan pengujian fitur baru. Ini membantu menghemat waktu dan memungkinkan deteksi bug atau masalah lebih awal. Dalam sistem yang besar, pengujian regresi dapat memakan waktu, sehingga pengaturan pengujian secara paralel sangat membantu dalam meningkatkan efisiensi.
5. **Evaluasi dan Dokumentasi Hasil:** Setelah pengujian regresi dilakukan, hasilnya harus dianalisis untuk memastikan tidak ada regresi atau penurunan kualitas fungsionalitas perangkat lunak. Jika ditemukan bug atau masalah baru, ini harus segera diatasi. Semua hasil pengujian harus didokumentasikan dengan baik untuk menjaga rekam jejak perbaikan dan perubahan pada perangkat lunak.
5. **Penggunaan CI/CD untuk Automasi Pengujian Regresi:** Dalam proses modern pengembangan perangkat lunak, alat CI/CD (Continuous Integration/Continuous Delivery) sering digunakan untuk secara otomatis menjalankan pengujian regresi setelah setiap commit atau perubahan kode baru. Jenkins, GitLab CI, dan Travis CI adalah beberapa alat yang mendukung eksekusi otomatis test case regresi setiap kali ada perubahan dalam kode.

Pengujian regresi adalah elemen krusial dalam siklus pengembangan perangkat lunak yang harus dilakukan secara teratur setiap kali ada perubahan kode. Dengan mengotomatisasi pengujian regresi dan memanfaatkan tool CI/CD, tim pengembangan dapat memastikan bahwa perangkat lunak tetap stabil dan bebas bug meskipun ada perubahan fitur atau peningkatan sistem.

10. Kesimpulan

Regression Testing adalah salah satu jenis pengujian perangkat lunak yang krusial, dilakukan untuk memastikan bahwa perubahan atau pembaruan kode tidak mempengaruhi fungsionalitas yang sudah ada dalam aplikasi. Pengujian ini penting setelah adanya perbaikan bug, penambahan fitur baru, atau peningkatan sistem. Tujuan utama dari regression testing adalah untuk menjaga stabilitas dan integritas perangkat lunak, sehingga fungsionalitas lama tetap bekerja dengan baik setelah perubahan dilakukan. Pengujian regresi dapat dilakukan secara manual atau otomatis tergantung pada skala dan frekuensi perubahan.

Terdapat berbagai jenis regression testing, seperti **Retest All**, **Regression Test Selection**, dan **Priority-based Regression Testing**, yang masing-masing memiliki kelebihan dan kekurangan. Retest All memastikan bahwa seluruh fitur diuji ulang, namun memerlukan banyak waktu dan sumber daya. Sementara itu, Regression Test Selection dan Priority-based Regression Testing berfokus pada area yang paling terpengaruh oleh perubahan kode, yang lebih efisien dari segi waktu. Automasi regression testing sangat dianjurkan karena dapat mengurangi waktu pengujian dan memastikan konsistensi dalam pengujian berulang.

Tools seperti **Selenium**, **Cypress**, dan **TestNG** sangat membantu dalam mengotomatisasi regression testing. Automasi memungkinkan pengujian dilakukan secara cepat dan efisien, terutama dalam skenario Continuous Integration/Continuous Delivery (CI/CD). Dengan menerapkan regression testing secara berkala dan otomatis, tim pengembang dapat mengidentifikasi potensi masalah lebih awal dan memastikan kualitas perangkat lunak tetap terjaga, bahkan dengan perubahan kode yang sering terjadi dalam siklus pengembangan modern.

11. Evaluasi Pembelajaran Regression Testing

Regression Testing merupakan komponen penting dalam siklus pengembangan perangkat lunak yang memastikan bahwa perubahan atau pembaruan pada kode tidak mengganggu fungsionalitas yang sudah ada. Pengujian ini membantu menjaga stabilitas dan kualitas aplikasi dalam jangka panjang. Dengan memahami regression testing, mahasiswa akan lebih mampu memahami teknik pengujian yang efektif untuk mempertahankan performa aplikasi secara keseluruhan setelah pembaruan atau perbaikan. Evaluasi ini akan menguji pemahaman mengenai konsep dasar, jenis-jenis pengujian regresi, dan implementasi praktisnya.

1. Apa perbedaan utama antara regression testing manual dan regression testing otomatis? Berikan situasi di mana masing-masing metode lebih efektif digunakan dalam pengujian perangkat lunak.
2. Jelaskan bagaimana pendekatan "Priority-based Regression Testing" dapat meningkatkan efisiensi dalam pengujian regresi. Sebutkan langkah-langkah yang harus diambil untuk menentukan prioritas area pengujian dan bagaimana ini berdampak pada hasil pengujian.
3. Diskusikan manfaat dari pengujian regresi otomatis dalam skenario pengembangan perangkat lunak berbasis Agile. Bagaimana regression testing otomatis berperan dalam mendukung strategi Continuous Integration/Continuous Delivery (CI/CD)?
4. Sebuah fitur baru telah ditambahkan ke aplikasi e-commerce. Jelaskan pendekatan regression testing yang dapat Anda gunakan untuk memastikan fitur ini tidak merusak fungsi-fungsi lain di aplikasi tersebut. Sertakan teknik atau alat yang Anda rekomendasikan dalam jawaban Anda.
5. Berikan contoh implementasi regression testing pada suatu sistem manajemen data. Jelaskan tahapan-tahapan yang akan Anda lakukan dalam proses regression testing, serta alat yang akan Anda gunakan. Jelaskan mengapa alat tersebut tepat untuk kebutuhan pengujian sistem ini.

Referensi:

1. Myers, G. J., Sandler, C., & Badgett, T. (2011). The Art of Software Testing (3rd ed.). John Wiley & Sons.
2. Galin, D. (2004). Software Quality Assurance: From Theory to Implementation. Pearson Education.
3. Katalon. (n.d.). What is Regression Testing? Retrieved from https://katalon.com.
4. Beizer, B. (1990). Software Testing Techniques (2nd ed.). Van Nostrand Reinhold.
5. Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.
6. Kaner, C., Falk, J., & Nguyen, H. Q. (1999). Testing Computer Software (2nd ed.). Wiley.
7. Crispin, L., & Gregory, J. (2009). Agile Testing: A Practical Guide for Testers and Agile Teams. Addison-Wesley.
8. Graham, D., Van Veenendaal, E., Evans, I., & Black, R. (2007). Foundations of Software Testing (2nd ed.). Cengage Learning.
9. Postman. (2022). Using Postman for Regression Testing. Retrieved from https://www.postman.com.
10. Cypress.io. (2022). Cypress Documentation: Regression Testing. Retrieved from https://www.cypress.io
11. Fowler, M., & Highsmith, J. (2001). The Agile Manifesto. Retrieved from https://agilemanifesto.org
12. Kumar, R., & Kaur, H. (2021). "A Study on Regression Testing Techniques and Tools." International Journal of Computer Applications, 174(1), 7-11.