SOFTWARE

QUALITY ASSURANCE

written by:

TEAM TWO

**KATA PENGANTAR**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

# **DAFTAR ISI**

[DAFTAR ISI 3](#_Toc117666418)

[DAFTAR GAMBAR 5](#_Toc117666419)

[BAB I](#_Toc117666420) [INTRODUCTION 6](#_Toc117666421)

[A. Software Development Life Cycle (SDLC) 6](#_Toc117666422)

[B. Testing Definition 8](#_Toc117666423)

[C. Software Defect (Bug) 10](#_Toc117666424)

[BAB II](#_Toc117666425) [TYPES OF TESTING 13](#_Toc117666426)

[A. User Interface Testing 13](#_Toc117666427)

[B. Functional Testing 13](#_Toc117666428)

[C. Compatibility Testing 15](#_Toc117666429)

[D. Localization Testing 15](#_Toc117666430)

[E. Performance Testing 15](#_Toc117666431)

[F. Usability Testing 16](#_Toc117666432)

[G. Security Testing 16](#_Toc117666433)

[H. Monkey Testing (Ad-hoc) 17](#_Toc117666434)

[BAB III](#_Toc117666435) [BUGS 18](#_Toc117666436)

[A. How to Report Bugs 18](#_Toc117666437)

[B. Bug Triage 20](#_Toc117666438)

[C. Severity and Priority 20](#_Toc117666439)

[D. Priority Classes 21](#_Toc117666440)

[E. Severity tips 21](#_Toc117666441)

[BAB IV](#_Toc117666442) [TEST CASES 23](#_Toc117666443)

[A. What is Test Case 23](#_Toc117666444)

[B. Test Case vs Bug 23](#_Toc117666445)

[C. Why Test Cases is Important 24](#_Toc117666446)

[D. How to Write Test Cases 25](#_Toc117666447)

[E. Smoke Testing 25](#_Toc117666448)

[F. How to Write Test Plan 26](#_Toc117666449)

[G. How to Write Smoke Suite? 26](#_Toc117666450)

[BAB V](#_Toc117666451) [CROSS-PLATFORM AND CROSS-BROWSER TESTING 28](#_Toc117666452)

[A. How to Test on Mobile Devices? 28](#_Toc117666453)

[B. How to Use Chrome Mobile View? 28](#_Toc117666454)

[C. Modern Browsers 29](#_Toc117666455)

[D. The Most Popular Browsers 30](#_Toc117666456)

[E. Browsers Segmentation 30](#_Toc117666457)

[F. How to Test Android Apps Without A Device 31](#_Toc117666458)

[G. How to Test on IOS Without A Device 31](#_Toc117666459)

[H. How to Test Using Cloud Services 32](#_Toc117666460)

[I. What is Cross-Browser Testing 33](#_Toc117666461)

[BAB VI](#_Toc117666462) [REGRESSION TESTING 34](#_Toc117666463)

[A. Test Plans 34](#_Toc117666464)

[B. What is Regression Testing? 37](#_Toc117666465)

[C. What is Positive and Negative Testing 38](#_Toc117666466)

[D. What is Boundary Value Analysis? 39](#_Toc117666467)

[E. What is Equivalence Partitioning Testing? 39](#_Toc117666468)

[BAB VII](#_Toc117666469) [WEB ARCHITECTURE 41](#_Toc117666470)

[A. Modern Web Architecture Explained 41](#_Toc117666471)

[B. Back-End Testing 41](#_Toc117666472)

[C. Front-End Testing 42](#_Toc117666473)

[D. How to Debug Front-End or Back-End Issues? 43](#_Toc117666474)

[BAB VIII](#_Toc117666475) [BUILD AND RELEASES 44](#_Toc117666476)

[A. What is Software Build? 44](#_Toc117666477)

[B. What is Deployment? 44](#_Toc117666478)

[C. What is Release? 45](#_Toc117666479)

[D. SDLC Recapitulation 46](#_Toc117666480)

[E. What is Sanity Testing? 46](#_Toc117666481)

[F. What is User Acceptance Testing? 47](#_Toc117666482)

# **DAFTAR GAMBAR**

[Gambar 1 Gambaran sederhana tentang SDLC 7](#_Toc117666248)

[Gambar 2 Kegiatan-kegiatan utama saat testing 8](#_Toc117666249)

[Gambar 3 Persentase pengaruh bug terhadap software 10](#_Toc117666250)

[Gambar 4 Ilustrasi pengujian black box 12](#_Toc117666251)

[Gambar 5 Shortcut tangkapan layar pada beberapa sistem operasi 15](#_Toc117666252)

[Gambar 6 Log bugs pada jira 16](#_Toc117666253)

[Gambar 7 Tampilan type pada jira 16](#_Toc117666254)

[Gambar 8 Tampilan summary pada jira 16](#_Toc117666255)

[Gambar 9 Tampilan description pada jira 17](#_Toc117666256)

[Gambar 10 Tampilan hasil screenshot pada jira 17](#_Toc117666257)

[Gambar 11 Tampilan attachment hasil screenshot pada jira 17](#_Toc117666258)

# **BAB I**

# **INTRODUCTION**

1. **Software Development Life Cycle (SDLC)**

SDLC (Software Development Life Cycle) adalah kerangka kerja atau model manajemen proyek terstruktur yang menguraikan fase-fase yang diperlukan untuk membangun sistem TI, dari awal hingga hasil akhir. Tujuan dari SDLC ini untuk menciptakan proses produksi yang efektif dan berkualitas tinggi yang memenuhi harapan pelanggan dalam anggaran dan jadwal yang ditentukan. SDLC memiliki cara kerja dengan mengurangi biaya pengembangan perangkat lunak, meningkatkan kualitas dan mempersingkat waktu produksi. SDLC juga dapat membantu tim mencapai tujuan mereka dengan mengevaluasi kekurangan sistem.

Berikut penjelasan sederhana tentang SDLC, yang dianalogikan dengan pengalaman restoran. Ada pelanggan yang memesan sesuatu yang dinginkan, tapi cara bicara pelanggan dan cara memasak chef memiliki bahasa yang berbeda. Oleh karena itu, dibutuhkan penerjemah bahasa yang disebut sebagai pemilik produk. Dalam kasus restoran ini, dapur adalah pengembangan tim (development tim). Juru masak (chef) sebagai pengembang yang melakukan sesuatu yang mana chef ini yang dapat merasakan terlebih dahulu sesuatu yang mereka masak. Jadi, sebelum makanan dari chef tersebut di hidangkan ke pelanggan, maka hanya chef yang mengetahui rasa dan penampilan dari makanan tersebut yang sesuai dengan permintaan pelanggan. Apabila, pesanan pelanggan sudah sesuai dengan keinginannya dan kualitas cukup baik untuk dilayani, maka akan dihidangkan ke pelanggan. Berikut gambaran dari SDLC sederhana yang dianalogikan sebagai restoran:



Gambar 1 Gambaran sederhana tentang SDLC

1. **Testing Definition**

Untuk memastikan kualitas dari suatu aplikasi, testing harus dilakukan yaitu dengan cara menguji apakah sistem data yang dihasilkan sesuai dengan testing yang telah dilakukan. Tujuan dari testing ini sendiri antara lain, yaitu:

1. Memastikan aplikasi berjalan sebagaimana mestinya
2. Mendeteksi terjadinya error serta memvalidasi apakah sudah memenuhi keinginan user
3. Melakukan pengecekan/pengetesan entitas-entitas, termasuk software,untuk pemenuhan dan konsistensi dengan melakukan evaluasi hasil terhadap kebutuhan yang telah ditetapkan
4. Melakukan validasi untuk dapat melihat kebenaran sistem, apakah proses yang telah ditulis dalam spesifikasi adalah apa yang sebenarnya.
5. Mendeteksi adanya error, testing untuk mendeteksi kesalahan secara insentif, yaitu menentukan apakah suatu hal tersebut terjadi bilamana tidak seharusnya terjadi.

Terdapat dua tahapan proses perangkat lunak bebas bug dan memenuhi semua persyaratan, yaitu:

1. Verifikasi

Verifikasi adalah suatu pengecekan ataupun pengetesan entitas-entitas, termasuk aplikasi, untuk pemenuhan serta konsistensi dengan melakukan penilaian hasil terhadap kebutuhan yang telah ditetapkan.

“are we building the product right?” (menurut Boehm)

“apakah kita telah membuat produk dengan benar?”

1. Validasi

Validasi yaitu setelah melaksanakan proses verifikasi berikutnya adalah melihat kebenaran sistem, apakah proses yang ditulis dalam spesifikasi adalah apa yang sesungguhnya diinginkan ataupun dibutuhkan oleh pengguna.

“are we building the right product?” (menurut Boehm)

“apakah kita telah membuat produk yang benar?”

Berikut 5 kegiatan utama yang akan dibahas pada testing definition, yaitu:



Gambar 2 Kegiatan-kegiatan utama saat testing

1. Bug Hunting, yaitu aktivitas untuk mencari bug dan ketika bug sudah ditemukan akan dilaporkan.
2. Bug reporting, yaitu pemantauan dalam melacak status dan riwayat, yang mana memerlukan tempat pusat dalam proses pemantauan.
3. Prioritization, yaitu memprioritaskan perbaikan pada kekurangan yang masuk akal saja karena tidak semuanya akan diperbaiki.
4. Test cases, yaitu menulis, mengeksekusi, dan melaporkan kasus yang akan diuji.
5. Story testing, yaitu aktivitas untuk melakukan beberapa pengujian terhadap persyaratan, sehingga dapat menemukan bug, melaporkan dan menulis kasus yang diuji.
6. **Software Defect (Bug)**

Bug adalah kesalahan dalam program komputer yang menyebabkan hasilnya tidak benar atau tidak terduga, atau berperilaku dengan cara yang tidak diinginkan. Berikut ini adalah beberapa jenis bug yang paling sering terjadi selama proses pengembangan software.

1. Functional error

Functional error merupakan sebuah kategori luas yang mencakup masalah yang terkait fungsionalitas sebuah program. Bug jenis ini bervariasi, mulai dari tombol yang tidak dapat di-klik hingga masalah pada kegunaan aplikasi itu sendiri.

1. Performance defects

Performance defects adalah kategori bug yang terkait dengan kecepatan, stabilitas, response time, dan penggunaan sumber daya dari sebuah software.

1. Usability defects

Usability defects adalah jenis bug yang menyebabkan pengguna tidak dapat memanfaatkan sebuah software secara maksimal. Jenis bug ini biasanya menyebabkan software sulit atau tidak nyaman untuk digunakan. Selain masalah pada kode software, usability defects juga dapat disebabkan oleh desain UI yang terlalu rumit sehingga pengguna kesulitan menemukan fungsi yang mereka cari.

1. Compatibility error

Compatibility error merujuk pada masalah software yang tidak dapat berjalan sebagaimana mestinya dalam situasi tertentu. Pada umumnya, compatibility error akan muncul ketika dijalankan aplikasi lama pada sistem operasi yang lebih baru.

1. Security error

Security error merupakan jenis bug yang cukup berbahaya karena terkait langsung dengan sistem keamanan sebuah software. Sebuah bug dalam sistem keamanan software, besar atau kecil, akan membuka celah untuk cracking dan mencuri data atau merusak software tersebut.

1. Syntax error

Syntax error adalah jenis bug yang terjadi pada source code sebuah program. Jenis bug ini akan menghambat proses pengembangan software dan menyebabkan software tidak dapat di-compile dengan baik.

1. Logic error

Logic error adalah jenis bug lain yang terjadi pada source code sebuah program. Jenis bug ini biasanya disebabkan oleh penggunaan kode yang tidak sesuai sehingga menghasilkan output yang salah.



Gambar 3 Persentase pengaruh bug terhadap software

Pada gambar 3 dijelaskan bahwa functionality merupakan hal utama yang menjadi masalah /defect yang sangat berpengaruh terhadap software.

# **BAB II**

# **TYPES OF TESTING**

1. **User Interface Testing**

Kerusakan/ketidaksempurnaan pada UI/design dapat dikategorikan dalam 5 kategori, yaitu:

1. Layout

Kerusakan pada layout dapat terlihat pada ketidaksejajaran margin, ketidaksempurnaan model kotak IE6(Internet Explorer 6), overlapping, kerusakan gambar, jarak lebar dan tinggi garis, ruang yang tidak diingankan diantara daftar item, jarak vertikal yang memiliki kecacatan.

1. Font

Kerusakan pada font dapat terlihat dari tipe font yang berbeda, baik dari family font yang digunakan maupun ukuran font yang digunakan.

1. Color

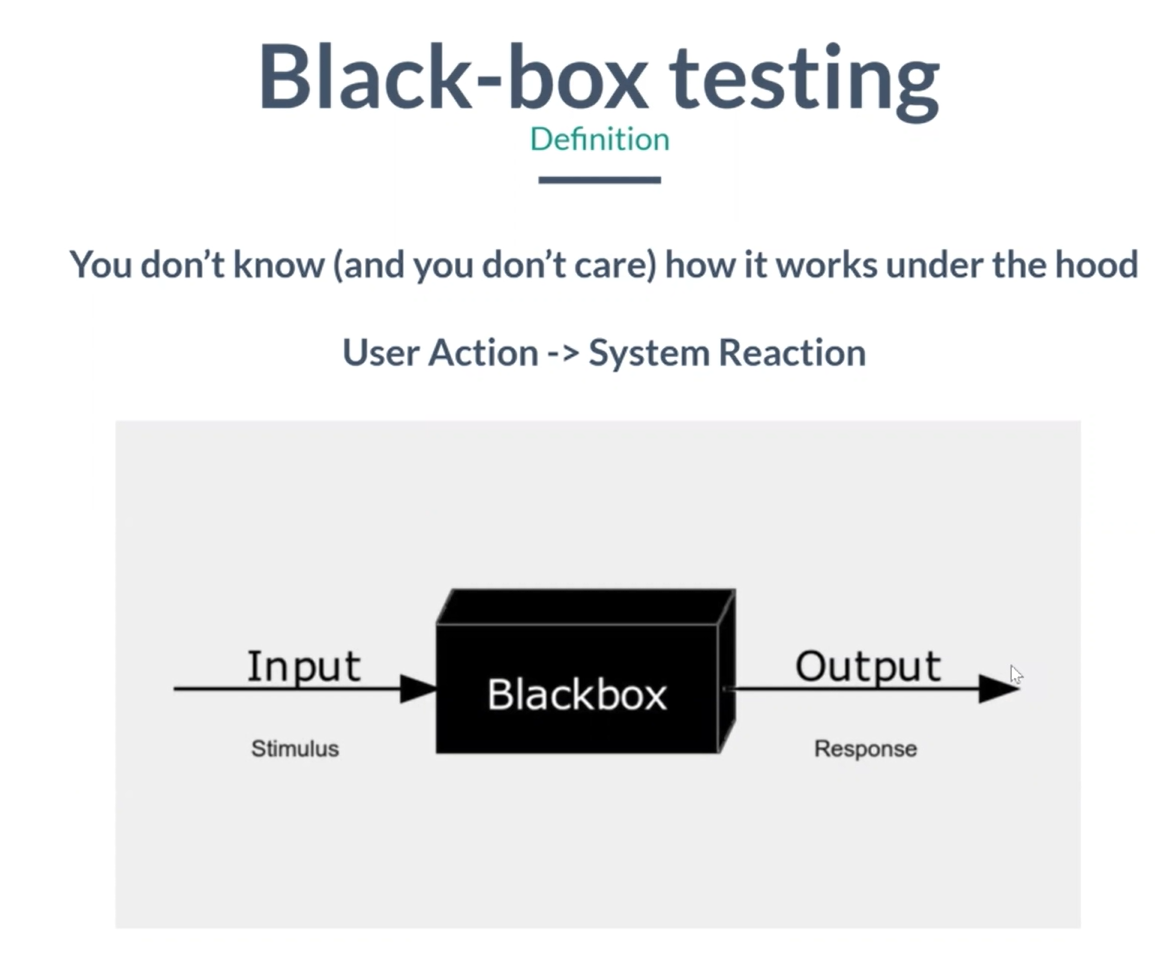
Kerusakan pada warna dapat terlihat dari tidak adanya keseragaman warna yang digunakan dan warna tombol yang tidak berfungsi sesuai dengan kebutuhan.

1. Content

Kerusakan pada content dapat terlihat dari perbedaan besar kecilnya suatu ukuran font, tidak adanya keseragaman kata, kerusakan gambar serta kesalahan dalam ejaan kata.

1. **Functional Testing**

Fungsional testing adalah sistem yang diuji terhadap fungsional persyaratan atau spesifikasinya. Jenis pengujian ini juga memverifikasi bahwa setiap fungsi aplikasi telah beroperasi sesuai dengan kebutuhan, yang berfokus pada manual dan automation testing. Fungsional testing ini merupakan salah satu jenis pengujian metode black box testing di mana setiap fungsionalitas aplikasi diuji dengan memberikan satu set input, yang diberikan untuk mengetahui reaksi aplikasi yang sebenarnya dan kemudian membandingkannya dengan hasil yang diharapkan sesuai dengan spesifikasi yang diberikan. Berikut contoh gambaran dari black box testing:



Gambar 4 Ilustrasi pengujian black box

Pengguna memberikan aksi sebagai input dan sistem memberikan reaksi/respon sebagai output. Black box testing hanya akan menjangkau input dan output sistem software tanpa adanya pengetahuan terkait internal program. Metode pengujian black box ini sangat penting agar bisa menemukan gangguan atau bug dalam suatu aplikasi sebelum di publikasikan secara resmi. Adapun beberapa jenis dari functional testing, yaitu:

1. Unit Testing: Pengujian secara individual dan independen dari bagian aplikasi terkecil yang dapat diuji.
2. Smoke Testing: Pengujian ini dilakukan pada pembuatan perangkat lunak awal untuk memeriksa apakah fungsi kritis berfungsi dengan baik atau tidak untuk melakukan pengujian lebih lanjut.
3. Sanity Testing: Pengujian yang dilakukan pada perangkat lunak yang stabil, yang mengalami perubahan kecil dalam kode atau fungsionalitas. Tujuannya adalah untuk memastikan bahwa kesalahan telah diselesaikan dan memastikan bahwa tidak ada lagi kesalahan/bug yang disebabkan oleh perubahan baru.
4. Integration Testing: Pengujian yang dilakukan untuk memeriksa fungsi perangkat lunak yang benar ketika komponen digabungkan sebagai satu unit.
5. Regression Testing: Pengujian Regresi dilakukan ketika bug diperbaiki atau fungsionalitas baru telah diperkenalkan ke suatu program. Ini memastikan bahwa fungsionalitas yang ada tidak rusak karena pengenalan perubahan baru.
6. User Acceptance Testing: ini adalah jenis pengujian perangkat lunak di mana suatu sistem diuji untuk penerimaan pelanggan. Satu-satunya tujuan pengujian ini adalah untuk mengevaluasi kepatuhan sistem dengan persyaratan bisnisnya dan untuk mengevaluasi apakah dapat diterima untuk pengiriman. Ini juga memverifikasi apakah perangkat lunak dapat melakukan tugas tertentu dalam situasi nyata.
7. **Compatibility Testing**

Compatibility testing merupakan bagian dari non-fungsional testing pada perangkat lunak, yaitu pengujian yang dilakukan pada aplikasi untuk mengevaluasi kompatibilitas aplikasi dengan komputasi. Komputasi ini mencakup beberapa aspek seperti hardware, sistem operasi, database, browser, dan sebagainya. Compatibility testing digunakan untuk memeriksa apakah sistem yang dikembangkan mampu berjalan pada hardware, software, sistem operasi, atau pada jaringan yang berbeda. Tidak hanya itu, compatibility testing juga digunakan untuk mengetahui dan menentukan versi minimum untuk dapat menjalankan aplikasi pada perangkat yang sedang dilakukan uji coba. Semakin sistem dapat dijalankan di banyak jenis perangkat yang berbeda, maka semakin baik aspek kompatibilitasnya.

1. **Localization Testing**

Pengujian lokalisasi yaitu memeriksa aplikasi sesuai dengan setting lokal untuk menyesuaikan bahasa dan negara yang sesuai dengan keinginan pengguna. Oleh karena itu, format pada mata uang, tanggal dan waktu akan berubah sesuai dengan pengaturan lokal yang dipilih oleh pengguna.

1. **Performance Testing**

Performance testing yaitu untuk menentukan kecepatan akses, daya tanggap, dan stabilitas program perangkat lunak. Pada pengujian ini, seorang software tester / QA tidak dituntut untuk mencari error melainkan mencari hambatan-hambatan yang dapat membuat sebuah aplikasi loading terlalu lama atau mungkin gagal diakses. Aplikasi yang loading-nya lama tentu akan memberikan penilaian yang buruk pada pengguna. Adapun tujuan dari Performance testing yaitu mengevaluasi output aplikasi, kecepatan pemrosesan, kecepatan transfer data, penggunaan bandwidth jaringan, pengguna bersamaan maksimum, pemanfaatan memori, efisiensi beban kerja, dan waktu respon perintah. Terdapat tiga faktor utama dalam memenuhi harapan/hasil dari performance testing, yaitu:

1. Speed: Melihat kecepatan sistem dalam memberikan respon dari setiap request.
2. Scalability: Menentukan berapa jumlah load maksimum yang dapat ditangani sistem.
3. Stability: menganalisis kondisi sistem saat diberikan beban yang bervariasi, apakah stabil atau tidak.
4. **Usability Testing**

Usability testing adalah metode yang digunakan untuk mengevaluasi seberapa mudah situs web menggunakan User Experience (UX) yang dilakukan pada produk yang fokus pada identifikasi masalah kegunaan, mengumpulkan data kualitatif dan kuantitatif, dan menentukan kepuasan dari pengguna produk. Umumnya usability test mengevaluasi persyaratan fungsional dan kualitas dari user interface. Usability test ini juga memiliki hubungan antara pengguna dengan sistem yang dilakukan untuk menentukan apakah fungsi telah sama seperti yang diharapkan dan apakah user interface membuat sistem dapat mudah digunakan.

Pengujian ini sering dilakukan untuk mendapatkan hasil yang cepat dalam meningkatkan interface dan mengkoreksi kesalahan dalam komponen perangkat lunak. Salah satu contoh usability box new away 2018, ketika semua orang dari negara bagian mendapatkan pesan peringatan hanya karena kesalahan seseorang meng-klik sesuatu yang salah dikomputer, karena masalah penggunannya yang mana memiliki dua tombol yang sama disamping satu sama lain. Oleh karena itu, lebih baik untuk menggunakan warna yang berbeda agar dapat membedakan antara satu tombol dengan tombol yang bersampingan.

1. **Security Testing**

Security testing yaitu jenis pengujian perangkat lunak yang dilakukan untuk mengidentifikasi kerentanan serta memastikan bahwa data dan sumber daya sistem di dalamnya sudah terlindungi dengan baik dari para penyusup. Pengujian ini dilakukan dengan tujuan untuk menemukan semua celah dan kelemahan sistem yang dapat mengakibatkan hilangnya informasi. Adapun tahapan-tahapan dari security testing, sebagai berikut:

1. Asessment, yaitu analisis keamanan untuk persyaratan dan periksa kasus penyalahgunaan
2. Design, yaitu analisis risiko keamanan untuk perancangan. Pengembangan Rencana Uji termasuk uji keamanan
3. Develop, yaitu panduan kode bersama dengan pengujian untuk unit dasar.
4. Testing, yaitu pengujian black box, pengujian statis dan dinamis dan pengujian white box, dan pemindaian kerentanan.
5. Deploy, yaitu pengujian penetrasi, pemindaian kerentanan
6. Monitor, yaitu analisis dampak patch, pengujian regresi dan tinjauan berkala terhadap perkembangan lebih lanjut.
7. **Monkey Testing (Ad-hoc)**

Ad-hoc testing yaitu salah satu cara untuk menemukan bug dalam pengujian yang mengacu pada beberapa tes yang dilakukan secara acak. Setelah bug ditemukan, maka akan dilakukan tangkapan layar yang mana cara dalam menangkap layar tergantung pada sistem yang digunakan. Berikut shortcut dalam tangkapan layar pada beberapa sistem:



Gambar 5 Shortcut tangkapan layar pada beberapa sistem operasi

# **BAB III**

# **BUGS**

1. **How to Report Bugs**

Untuk melaporkan bug dapat menggunakan Jira, berikut 4 langkah utama log bugs dalam Jira:



Gambar 6 Log bugs pada jira

1. Type

Pilih jenis masalah bug, yang mana tergantung pada proyek yang telah disiapkan di JIRA. Jenis masalah dapat berisi nilai berikut: bug, story, epic, task dan sebagainya untuk melaporkan kerusakan.



Gambar 7 Tampilan type pada jira

1. Summary

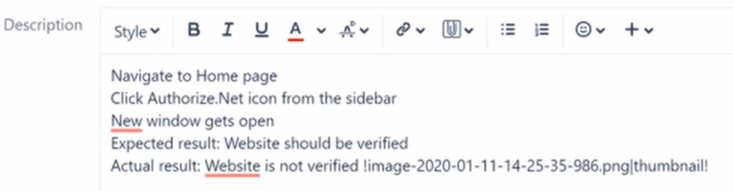
Menulis ringkasan singkat tentang apa dan dimana yang salah. Dapat ditulis di summary yaitu [Authorize.net] Verification is missing.



Gambar 8 Tampilan summary pada jira

1. Description

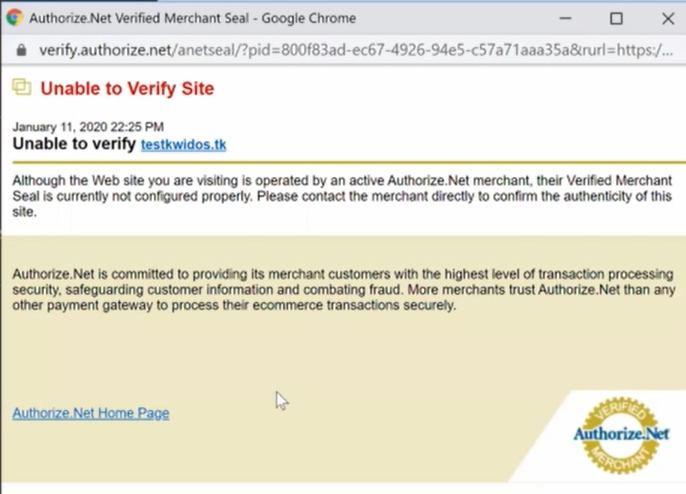
Deskripsi ini adalah bidang yang paling penting ketika membuat laporan bug, yang mana deskripsi ini sedikit lebih deskriptif dari dari ringkasan (summary).



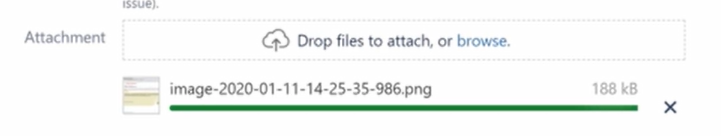
Gambar 9 Tampilan description pada jira

1. Screenshot

Screenshot jendela Authorize.Net untuk lebih terpercaya dan aman. Setelah itu, masukkan file image di attachment.



Gambar 10 Tampilan hasil screenshot pada jira



Gambar 11 Tampilan attachment hasil screenshot pada jira

1. **Bug Triage**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **Severity and Priority**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **Priority Classes**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **Severity tips**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

# **BAB IV**

# **TEST CASES**

1. **What is Test Case**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **Test Case vs Bug**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **Why Test Cases is Important**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **How to Write Test Cases**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **Smoke Testing**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **How to Write Test Plan**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **How to Write Smoke Suite?**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

# **BAB V**

# **CROSS-PLATFORM AND CROSS-BROWSER TESTING**

Cross-platform testing mengharuskan aplikasi harus berjalan sama baiknya (sebaiknya sempurna) di semua jenis komputer dan platform seluler dan/atau sistem operasi seperti Windows, OS, Android, Macintosh, Chrome OS, dan lain-lain. Ada banyak basis yang harus dicakup jika Anda ingin menjamin kinerja yang baik di semua platform potensial, aplikasi yang akan digunakan. Pengujian lintas platform menentukan semua kemungkinan masalah caching dan perilaku aplikasi seluler atau web pengguna di lingkungan dan perangkat yang berbeda. Ini membantu pengguna melihat produk perangkat lunak pengguna lebih luas dan mandiri, mencegah kerusakan fungsional, dan bahkan dapat meningkatkan kegunaan aplikasi secara keseluruhan.

1. **How to Test on Mobile Devices?**

Untuk melakukan pengujian dengan menggunakan perangkat seluler tentunya kamu harus:

1. Gunakan perangkat mobile sungguhan (ponsel)
2. Simulator/emulator pengguna (tampilan web chrome, install simulator atau emulator)
3. Layanan cloud pengguna untuk menyewa perangkat (SauceLabs, Browsertack)

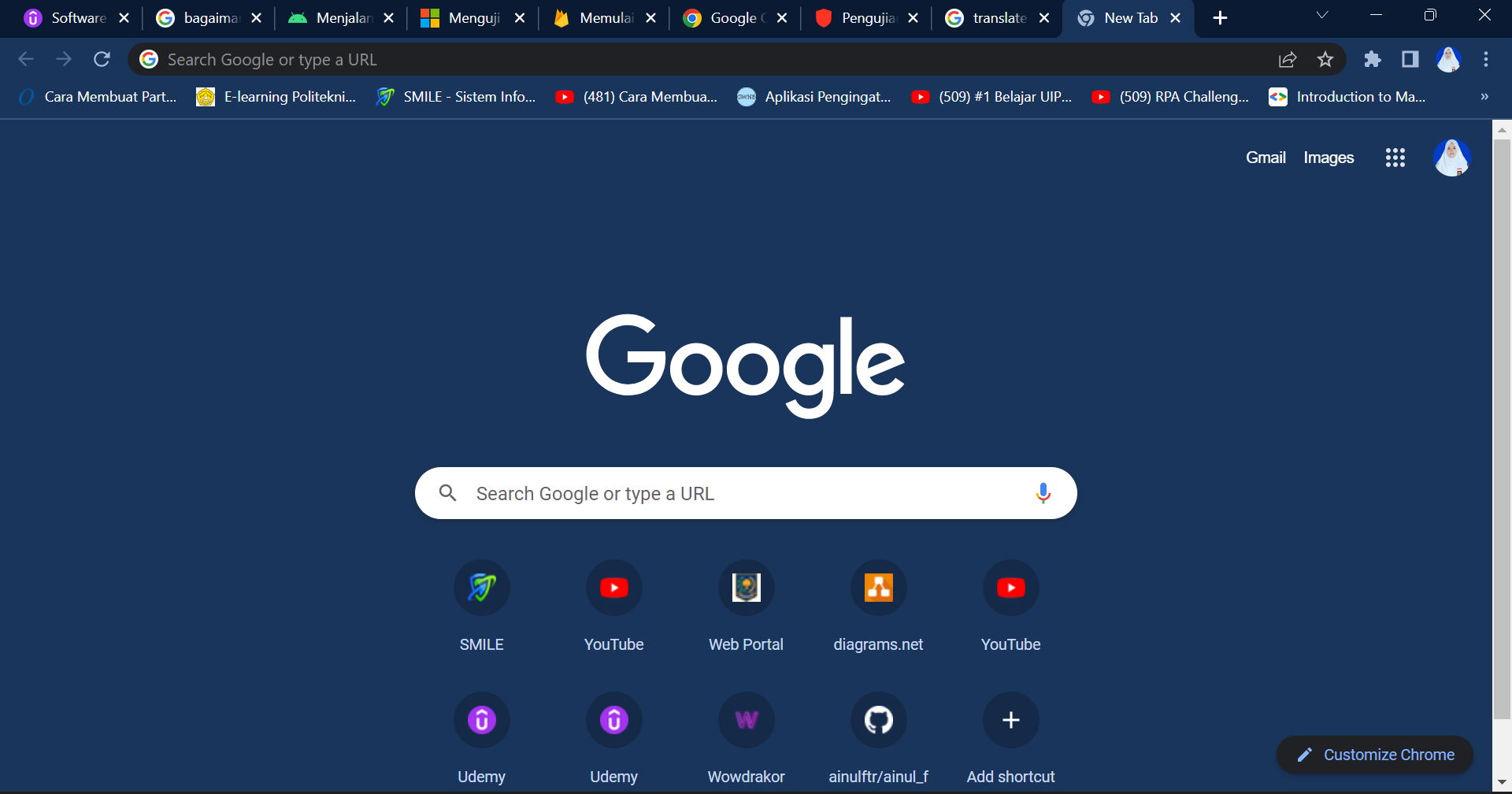
Bagaimana Anda dapat menguji di perangkat seluler. Tentu saja, menggunakan perangkatnya secara langsung, kita semua memiliki ponsel, apapun itu model, merk dan versiny. Tetapi tentu saja banyak sekali jenis yang tersedia di pasaran. Anda bisa menggunakan merek baru, bisa saja model lama, bisa android, OS dan sebagainya.

Masalahnya di sini adalah jika Anda ingin menguji pada perangkat partikular, itu hampir tidak mungkin karena, berapa banyak perangkat yang kita miliki? mungkin satu atau dua? Mungkin lebih atau tidak lebih dari itu. pilihan lainnya adalah menggunakan simulator atau emulator pengguna, salah satunya melalui browser Chrome. Dan pilihan lainnya ada layanan cloud khusus, jadi pengguna bisa membuat akun di sana, dan menggunakan komputer dasar melalui browser, pengguna dapat menavigasi ke situs web tersebut dan hanya memilih perangkat yang dibutuhkan seperti sistem operasi atau memilih versi.

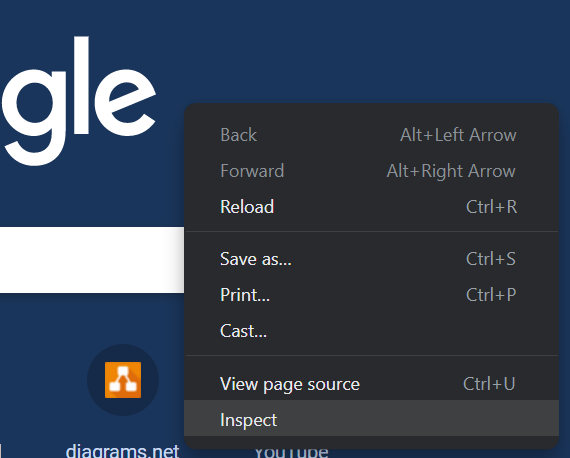
1. **How to Use Chrome Mobile View?**

Tampilan website mobile ditujukan untuk pengguna dengan perangkat mobile (smartphone & tablet), sementara website desktop adalah untuk perangkat PC(komputer desktop &laptop). Namun, bisakah kita mengakses tampilan website versi mobile menggunakan perangkat PC? Tentu saja bisa. Browser Google Chrome menyediakan sarana untuk membuka website mobile di komputer tanpa harus menginstal ekstensi tambahan. Disini akan ditunjukkan tampilan seluler di browser chrome, jadi browser Chrome adalah browser paling populer dan menarik. Jadi, misalnya kita akan menavigasi ke aplikasi kita yang sedang diuji. Berikut cara menggunakannya:

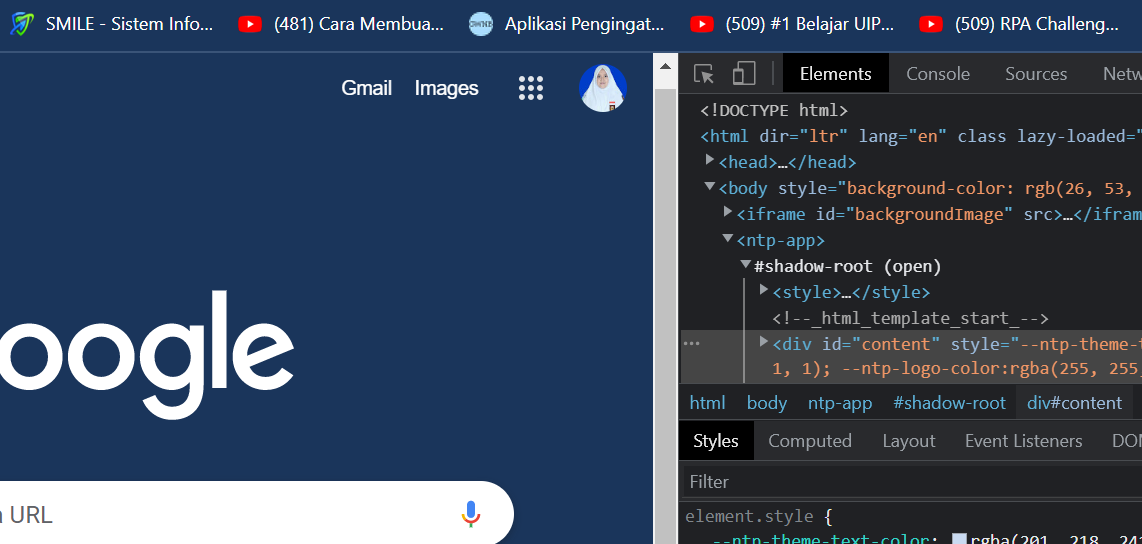
1. Buka website yang ingin kamu lihat versi mobile-nya di Google Chrome.



1. Klik kanan pada area mana saja di website tersebut kemudian klik Inspeksi (F12).

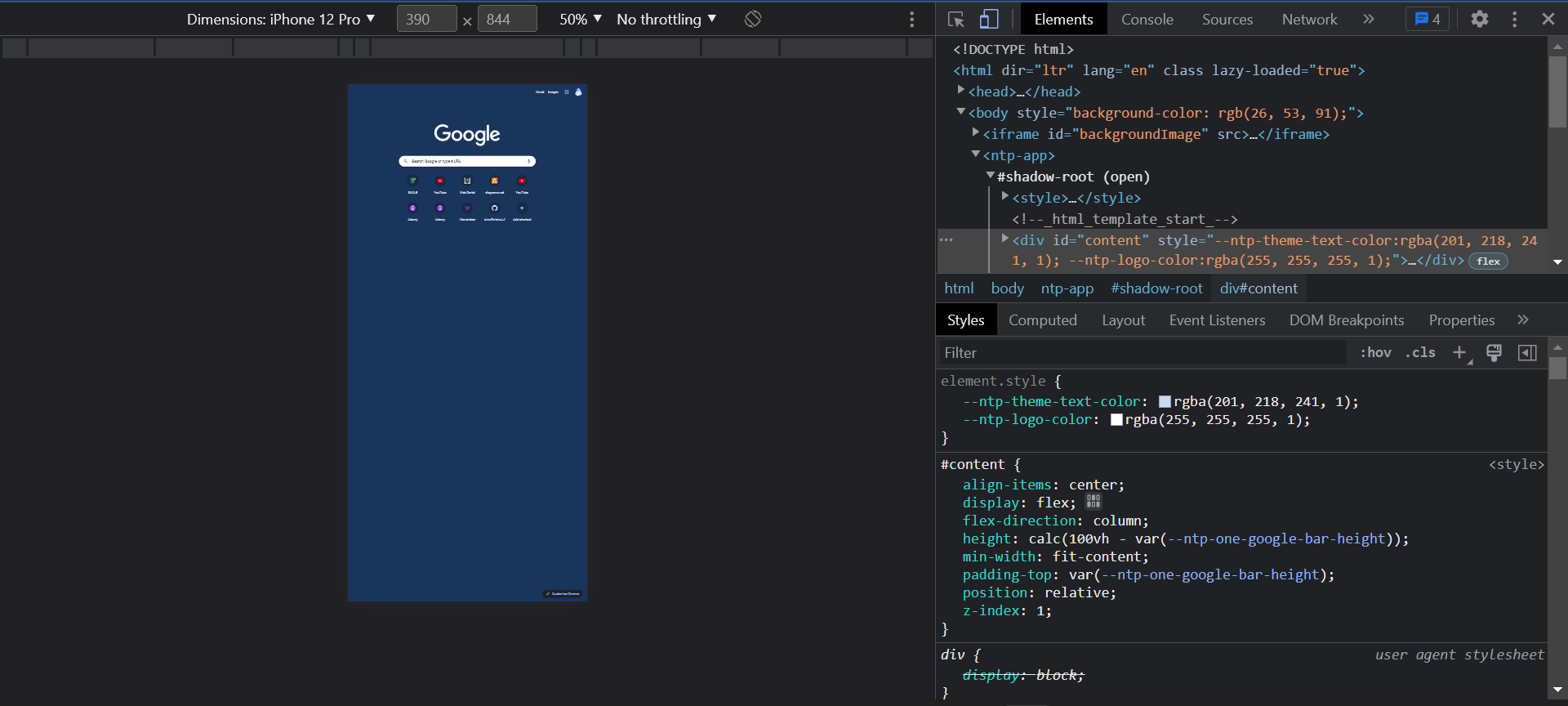


1. Akan muncul jendela baru di sebelah kanan, klik ikon Device Toolbar yang berbentuk seperti perangkat smartphone/tablet.

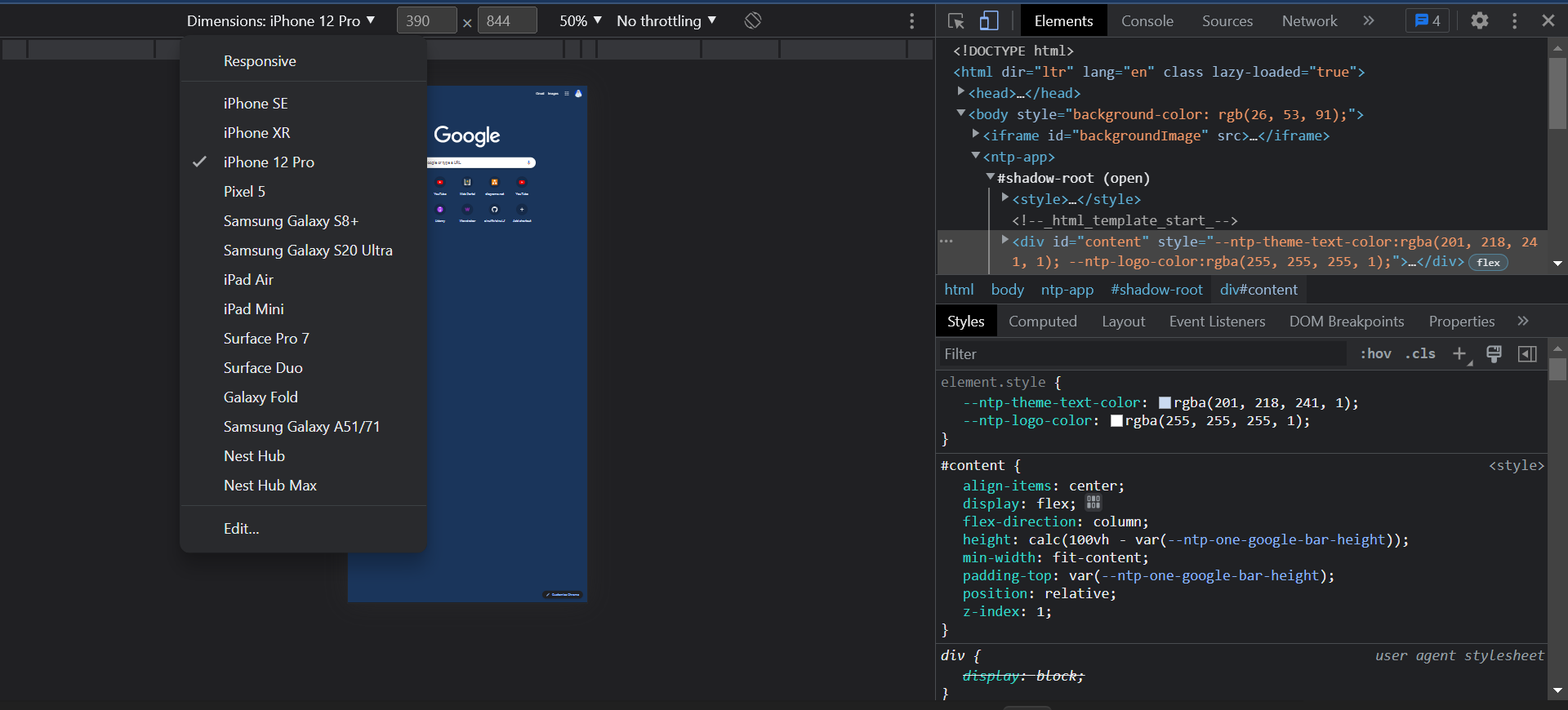


1. Kamu akan melihat jendela website jadi sedikit mengecil, namun ini masih merupakan tampilan versi desktop. Lihat langkah selanjutnya.

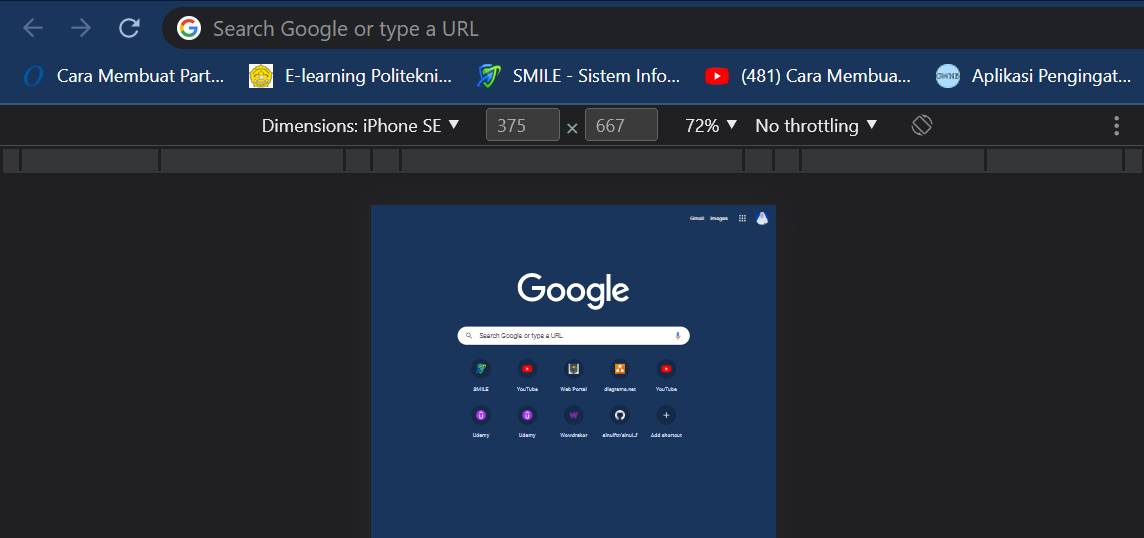




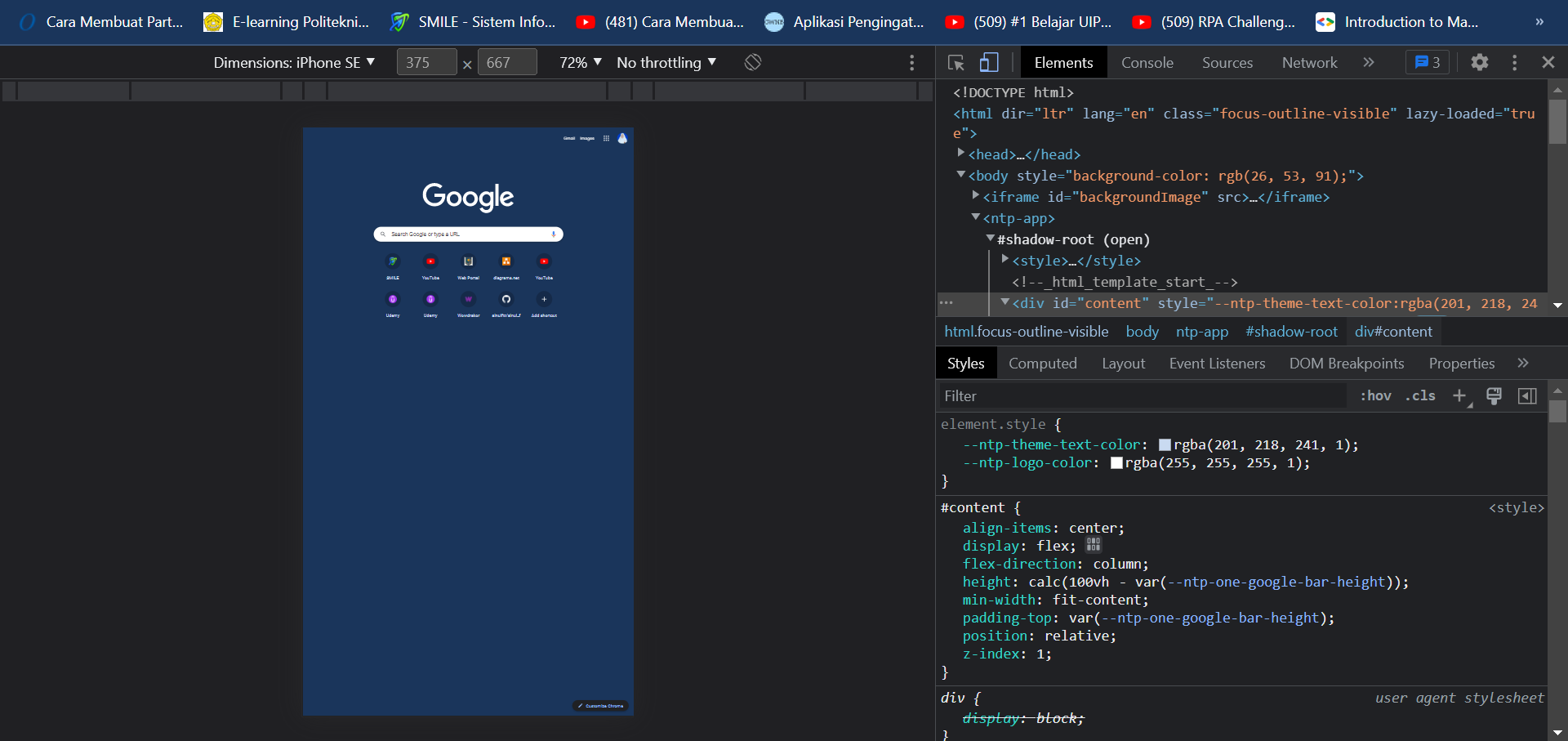
1. Pilih perangkat mobile apa yang ingin kamu jadikan patokan (berdasarkan ukuran layar). Kamu juga bisa mengatur dimensi layar secara manual.



1. Sekarang, muat ulang halaman (CTRL+R).



1. Sekarang tampilan website versi mobile sudah aktif.

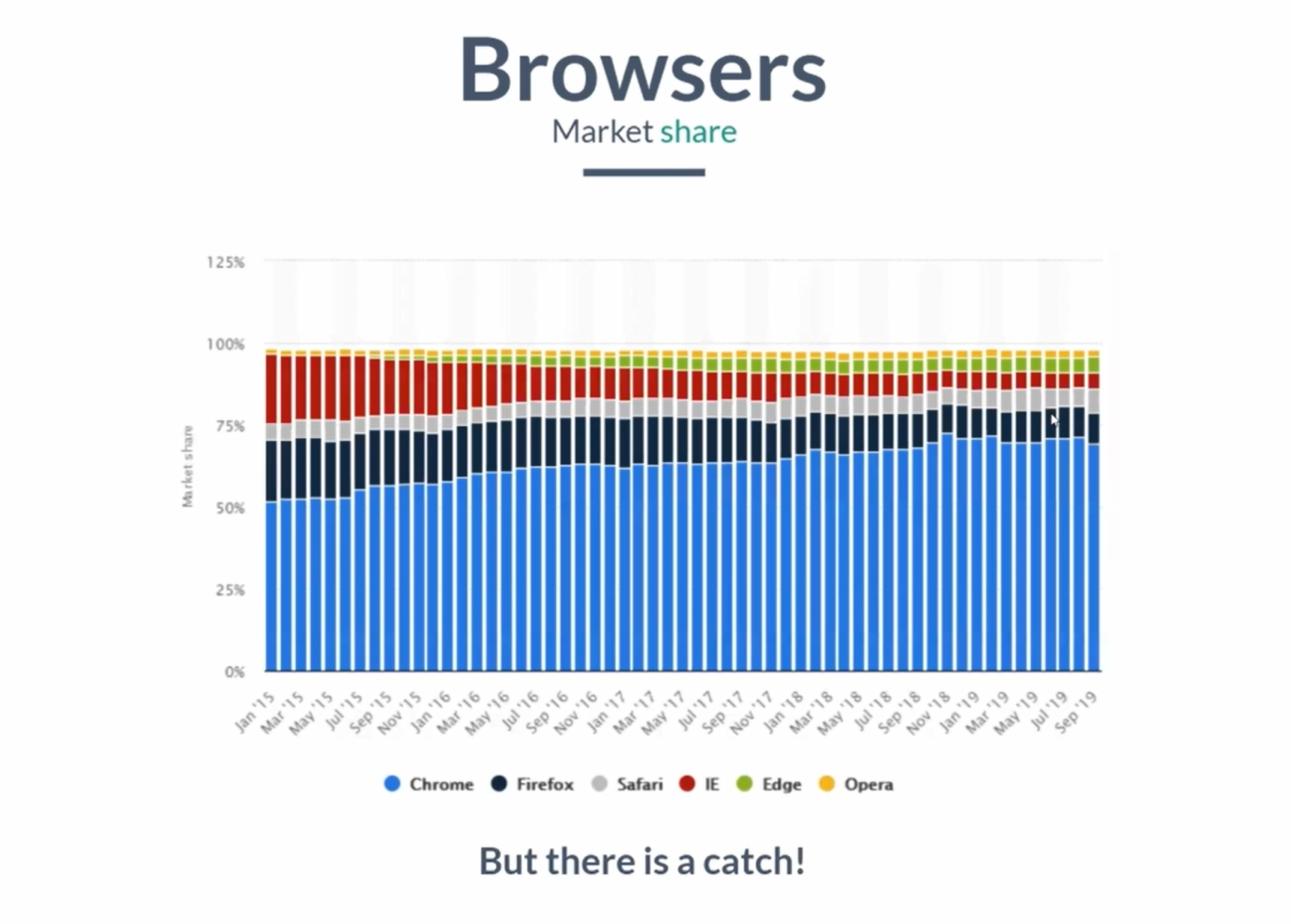


Tampilan mobile hanya muncul pada website yang dipilih dimana fitur Inspeksi dan Device Toolbar diaktifkan. Inspeksi ini bisa mematikan dan mengaktifkannya kapan saja saat dibutuhkan.

1. **Modern Browsers**

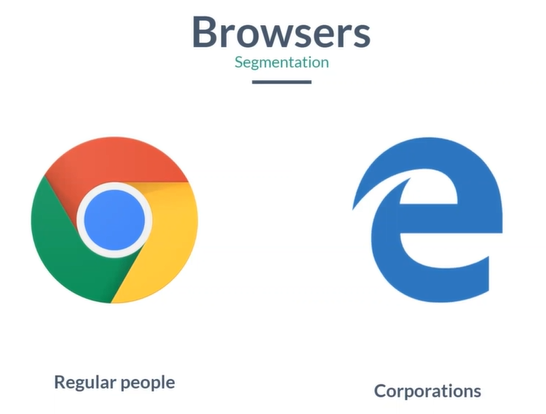
Jika ada pertanyaan browser mana yang seharusnya digunakan? Tentunya ada banyak browser yang dapat digunakan untuk ini, diantaranya terdapat Opera, Maxton dan banyak lainnya, namun Edge tidak terdaftar disini. Jadi jika bertanya browser apa yg dipilih untuk melakukan test? Dan jawabannya tidak seperti yang paling populer atau yang paling disukai. Itu harus yang digunakan pelanggan, dan pada kenyataannya, terkadang tidak seperti yang kita pikirkan, sebenarnya. Jadi ada istilah Market Share saat ini.

Browser Chrome pemegang peringkat tertinggi, ada sekitar 65, 70% dari pasar/pengguna, yang membuat browser ini menjadi yang paling populer tanpa sanggahan apapun. Kemudian ada firefox, dan dalam penurunan posisi dan safarinya. Egde menjadi sedikit lebih populer, tetapi opera masih memiliki kepercayaan, dua persen atau lebih. Beginilah tampilannya secara umum jika mempertimbangkan jenis seluruh internet, bisa dilihat di sini karena yang paling populer dan mana yang paling rendah peminatnya.



1. **Browsers Segmentation**

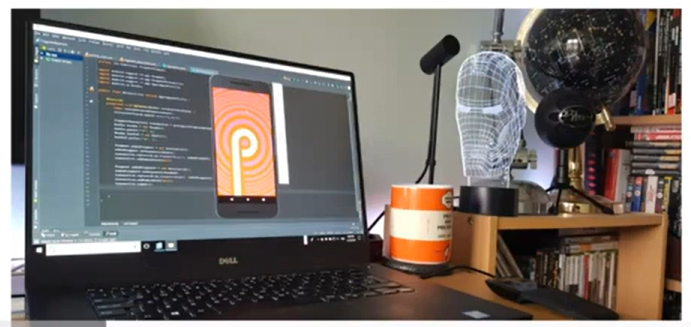
Berdasarkan hasil survei yang telah dilakukan, 70% orang di planet Bumi menggunakan Google Chrome. Namun, bisnis besar tidak selalu menjadikan Chrome sebagai browser pilihan mereka. Misalnya, pada Badan Pendataan Penduduk (Sensus) United States Federal Bereau Census Amerika Serikat, salah satu organisasi Federal, Ini adalah badan pemerintahan, Chrome ilegal jika digunakan untuk sensus. Jadi mereka adalah memilik menggunakan Microsoft Edge sebagai internet explorer pada saat itu. Dan tidak hanya itu, tapi sebagian besar perusahaan lainnya juga demikian memilih menggunakan internet explorer, sehingga Edge memiliki kedudukan sekitar 70% pengguna dipasarnya. Jadi terlepas dari kenyataan bahwa chrome sangat populer, internet explorer sebenarnya adalah browser pilihan dari banyak perusahaan besar dan dijadikan Browser nomor satu dan kemudian Chrome. kecuali dalam hal sensus, mereka tidak menggunakan Chrome melainkan memilih menggunakan dorefox karena chrome ilegal, mereka bahkan tidak diperbolehkan menginstal Chrome di komputer mereka, karena ilegal dan itu dilarang.



1. **How to Test Android Apps Without A Device**

Simulator dan emulator, jika kita ingin mendapatkan hasil yang menyerupai ke perangkat yang sebenarnya tanpa memiliki perangkat tersebut, kita dapat menginstal perangkat lunak (software) khusus di komputer dan beberapa di antaranya bisa didapatkan secara gratis, namun beberapa di antaranya tidak support android. Misalnya kita bisa menggunakan Google untuk emulator android. Kita dapat menemukan banyak fitur sehingga yang satu ini membentuk otoritas android, ada sekitar 15 Emulator terbaik ditahun 2022. Yang pertama ada Android Studio yang dapat dijadikan pilihan pertama atau bisa dikatakan rekomendasi.

Jika kita menggunakan perangkat seluler secara langsung dan gratis, namun setiap perangkat lunak itu perlu pengembangan. Dan dengan cara itulah pengembang Android membangun aplikasi seluler untuk android dan perangkat lunak (software) dan itu termasuk aplikasi yang berat. Ada banyak fitur dan banyak hal yang bisa dilakukan dalam Android Studio, jadi agak sedikit rumit untuk pengguna baru. Android Studio memiliki Emulator di Simulator, semuanya terpasang dalam Android Studio. Selanjutnya ada Jenny Motion, namun itu merupakan layanan berlangganan dan berbayar, tetapi bagaimanapun itu pasti mereka juga menyediakan layanan gratis dan kita dapat menggunakannya secara gratis. Ini seperti emulator namun lebih ringan dan kita bisa benar-benar melihat layar android seperti yang kita dilhat pada gambar dibawah ini.



Pada gambar diatas ada laptop dan terbuka jendela Aplikasi Sndroid Studio dan didalamnya ada gambar telepon genggam, dan kita dapat melakukan navigasi apa pun ke situs Web. Kita benar-benar dapat menggunakan aplikasi selulernya dan bermain-main dan mengubah beberapa hal. Kita juga dapat mengatur seperti G.P.S, dan lain-lain sebagainya.

1. **How to Test on IOS Without A Device**

Produk AS mirip seperti Andorid, hal yang sama kita dapat mengunduh somilator. dan yang paling populer adalah X code, tetapi ini diperuntukkan untuk pengembang Aplikasi. Begitulah cara pengembang A.S. membangun aplikasi asli, tapi aplikasi ini cukup berat dan memiliki banyak fitur, kita juga dapat menulis kode, dapat membuat aplikasi, dan ini berbayar. Kita harus berlangganan untuk menggunakannya, tetapi ada beberapa jenis lain yang tersedia.

1. **How to Test Using Cloud Services**

Layanan cloud, ada banyak hal yang menjadikan Cloud sangat populer. Jadi cara kerjanya, dan jika kita menonton video di youtube, itu akan memberi kita ide. kita dapat menggunakan semua fitur yang tersedia dalam cloud tersebut. tetapi singkatnya, emulator itu sebenarnya bisa selalu putar kembali atau diulang-ulang, tetapi harus melalui internet. kita hanya dapat menggunakannya di browser dan mereka memiliki banyak opsi, dan Cloud ini adalah layanan permium (berbayar). Layanan Could terkenal sangat mahal. Hal yang menarik dari layanan ini, kita tidak hanya dapat menggunakan emulator, tetapi kita benar-benar dapat menggunakan perangkat nyata yang terhubung melalui internet. Jadi seperti menggunakan perangkat nyata yang sebenarnya terletak disuatu tempat tapi kita dapat melihat lokasi perangkat juga. Itu tentunya jauh lebih mahal dan bisa saja itu berkisar antara $120 per bulan atau mungkin lebih.

1. **What is Cross-Browser Testing**

Pengujian lintas-browser secara lebih sempit mempunyai target pada fungsi kualitatif aplikasi/situs web untuk memberikan pengalaman yang luar biasa pada pengguna. Sebagian besar cara kerjanya di sini terkait dengan pengembang front-end yang harus memberikan solusi logis dan piksel sempurna.

Pengujian aplikasi ini memiliki peran besar dalam persepsi lebih lanjut tentang produk perangkat lunak Anda di pasar. Mereka biasanya bekerja baik pada browser paling banyak memiliki perubahan, dan pada sistem yang lebih lama. Pengujian lintas-browser memfasilitasi dalam mengidentifikasi masalah yang bervariasi dengan platform atau konfigurasi dan memastikan pertumbuhan dan kemakmuran bisnis pengguna berhasil.

# **BAB VI**

# **REGRESSION TESTING**

1. **Test Plans**

Test Plan adalah dokumen yang berisi definisi tujuan dan sasaran pengujian dalam lingkup iterasi (atau proyek), item-item yang menjadi target pengujian, pendekatan yang akan diambil, sumber daya yang dibutuhkan dan point untuk diproduksi. Dengan kata lain test plan dapat disebut sebagai perencanaan atau scenario untuk melakukan testing yang akan dilakukan baik oleh expert atau user umum. Tujuan membuat test plan secara umum adalah untuk memudahkan developer untuk melakukan testing agar testing yang dilakukan menjadi jelas sehingga hasilnya lebih berguna dan efisien.

Pembuatan test plan dapat dibuat dengan mengikuti template pembuatan test plan, namun tidak harus mengacu kepada template test. Berikut adalah penjelasan mengenai template Test Plan yang dikeluarkan oleh IEEE 829 (Institute of Electrical and Electronics Engineers) :

1. Test Plan Identifier

Test Plan Identifier adalah bagian untuk menjelaskan secara singkat mengenai objek yang akan di test. Bisa berupa penjelasan narasi atau berbentuk tabel dengan kategori kategori tertentu. Informasi yang dijelaskan dapat berupa sekilas mengenai subjek testing, nama orang yang bertanggung jawab terhadap testing, penyusun test plan , tanggal dibuat test plan dan tanggal revisi,dll.

1. Introduction

Pada bagian introduction dibuat untuk menjelaskan secara narasi, mengenai testing yang akan dilakukan terhadap suatu objek testing. Bagian Introduction dapat dibuat lebih rinci dengan menambahkan sub bab apabila perlu untuk dibuat. Contoh subbab yang dapat dibuat antara lain :

1. Purpose : untuk menjelaskan tujuan testing secara spesifik.
2. Background : latar belakang mengapa testing dilakukan.
3. Scope : Sejauh mana testing dilakukan.
4. Definition and Acronyms : penjelasan mengenai singkatan dan isitlah yang ada di dalam dokumen test plan.
5. Test Items

Bagian test item menjelaskan mengenai daftar komponen komponen dalam objek testing yang akan di test satu per satu.

1. Features to be tested

Penjelasan dan daftar daftar fitur yang akan ditest di pada saat pelaksanaan testing dimulai.

1. Features Not to be Tested

Menjelaskan mengenai fitur fitur apa saja yang ada di dalam objek testing namun, fitur tersebut tidak akan di test pada saat pelaksanaan testing dan disertakan penjelasan singkat mengapa fitur tersebut tidak di test pada saat testing.

1. Approach / Test Strategy

Bagian Approach adalah bagian yang digunakan untuk memberi deskirpsi mengenai Cara/approach yang dilakukan untuk melaksanakan testing dan disertakan dengan penjelasan mengenai approach yang digunakan.

1. Item Pass / Fail Criteria

Berisi tentang Kriteria- kriteria yang harus dipenuhi sebelum berlanjut ke fase berikutnya. Dan Kriteria Testing disebut gagal/fail.

1. Suspension Criteria

Berisi tentang Spesifikasi Kriteria- kriteria yang dapat digunakan untuk menghentikan sementara kegiatan testing dan testing tersebut dapat dilanjutkan di waktu lain.

1. Test Deliverables

Adalah List dokumen- dokumen apa saja yang akan dihasilkan setelah testing selesai dilakukan.

1. Testing Task

Menjelaskan Kegiatan testing beserta dengan pihak yang akan melaksanakan kegiatan/task tersebut.

1. Enviromental Needs

Spesifikasi dan Perincian segala sesuatu yang dibutuhkan dan digunakan selama proses testing berjalan, bisa berupa hardware yaitu spesifikasi komputer atau hal lain selain hardware.

1. Responsibilities

Rincian Pihak pihak yang akan bertanggung jawab terhadap suatu kegiatan task di dalam serangkaian kegiatan testing yang akan dilaksanakan.

1. Staffing and Training Needs

Secara garis besar menjelaskan bagaimana melakukan pendekatan untuk menentukan peran para staff di dalam proyek. Dan melakukan training apabila diperlukan untuk testing.

1. Schedule

Ada beberapa tujuan dalam membuat schedule di dalam test plan, antara lain :

1. Merincikan Tolak ukur waktu pengerjaan Testing.
2. Merincikan event transmittal item.
3. Estimasi waktu yang dibutuhkan untuk setiap task.
4. Menjadwalkan Testing task dan Test Milestone.
5. Merincikan periode pemakaian Testing resources.
6. Risk and Contingencies

Digunakan untuk memastikan agar hasil testing tetap berkualitas dengan memeriksa beberapa bagian yang tidak termasuk di dalam control pengerjaan software, namun bagian tersebut dapat berdampak langsung terhadap proses.

1. Approvals

Lembar persetujuan sebagai tanda bahwa seluruh tim/pimpinan telah menyetujui Test Plan yang telah dibuat.

1. **What is Regression Testing?**

Pengujian Regresi didefinisikan sebagai jenis pengujian perangkat lunak untuk mengonfirmasi bahwa program atau perubahan kode baru-baru ini tidak berdampak buruk pada fitur yang ada. Pengujian Regresi tidak lain adalah seleksi penuh atau sebagian dari kasus uji yang sudah dieksekusi yang dieksekusi ulang untuk memastikan fungsionalitas yang ada berfungsi dengan baik. Pengujian ini dilakukan untuk memastikan bahwa perubahan kode baru tidak memiliki efek samping pada fungsionalitas yang ada. Ini memastikan bahwa kode lama masih berfungsi setelah perubahan kode terbaru selesai.

Kebutuhan Pengujian Regresi terutama timbul setiap kali ada kebutuhan untuk mengubah kode dan kita perlu menguji apakah kode dimodifikasi mempengaruhi bagian lain dari aplikasi perangkat lunak atau tidak. Selain itu, pengujian regresi diperlukan, ketika fitur baru ditambahkan ke aplikasi perangkat lunak dan untuk perbaikan cacat serta perbaikan masalah kinerja.

Untuk melakukan proses Regression Testing , pertama-tama kita perlu men-debug kode untuk mengidentifikasi bug. Setelah bug diidentifikasi, perubahan yang diperlukan dibuat untuk memperbaikinya, kemudian pengujian regresi dilakukan dengan memilih kasus uji yang relevan dari rangkaian pengujian yang mencakup bagian kode yang dimodifikasi dan terpengaruh.

Pemeliharaan perangkat lunak adalah kegiatan yang meliputi peningkatan, koreksi kesalahan, pengoptimalan, dan penghapusan fitur yang ada. Modifikasi ini dapat menyebabkan sistem bekerja dengan tidak benar. Oleh karena itu, Pengujian Regresi menjadi perlu. Seleksi Uji Regresi adalah teknik di mana beberapa kasus uji yang dipilih dari rangkaian uji dijalankan untuk menguji apakah kode yang dimodifikasi mempengaruhi aplikasi perangkat lunak atau tidak.

Kasus uji dikategorikan menjadi dua bagian, kasus uji yang dapat digunakan kembali yang dapat digunakan dalam siklus regresi selanjutnya dan kasus uji usang yang tidak dapat digunakan pada siklus berikutnya. Prioritaskan kasus uji tergantung pada dampak bisnis, fungsi penting & sering digunakan. Pemilihan kasus uji berdasarkan prioritas akan sangat mengurangi rangkaian uji regresi.

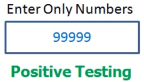
Ditemukan dari data industri bahwa sejumlah besar cacat yang dilaporkan oleh pelanggan disebabkan oleh perbaikan bug menit terakhir yang menciptakan efek samping dan karenanya memilih kasus uji untuk pengujian regresi adalah seni dan tidak semudah itu. Uji Regresi Efektif dapat dilakukan dengan memilih kasus uji berikut:

1. Kasus uji yang sering mengalami cacat
2. Fungsionalitas yang lebih terlihat oleh pengguna
3. Uji kasus yang memverifikasi fitur inti produk
4. Uji kasus Fungsionalitas yang telah mengalami lebih banyak dan perubahan baru-baru ini
5. Semua Kasus Uji Integrasi
6. Semua Kasus Uji Kompleks
7. Kasus uji nilai batas
8. Contoh kasus uji yang berhasil
9. Contoh kasus uji Kegagalan
10. **Positive Testing and Negative Testing**

Pengujian secara positif adalah untuk menentukan bahwa aplikasi berfungsi seperti yang diharapkan. Jika terjadi kesalahan selama pengujian positif, pengujian dianggap gagal. Adapun pengujian secara negatif adalah memastikan bahwa aplikasi dapat menangani input yang tidak valid atau perilaku pengguna yang tidak terduga.

1. **Positive testing**

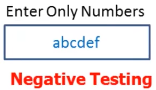
Pengujian ini berupa menguji inputan nilai yang valid. Kemudian melihat apakah aplikasi berjalan sesuai dengan harapan. Contohnya pada inputan berikut ini:



Memasukkan nilai dengan 99999 pada kolom tersebut dapat diterima/valid oleh sistem dikarenakan kolom tersebut bertipe Number.

1. **Negative testing**

Pada negative testing, pengujian yang dilakukan berupa menguji inputan yang salah, dan melihat apakah program akan memberikan notifikasi berupa invalid atau tidak. Contohnya jika kolom inputan seperti berikut :

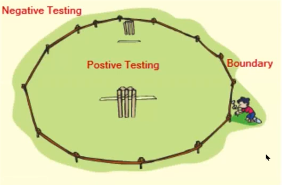


Menginput nilai selain dari angka seperti nilai A-Z/a-z pada kolom input number maka aplikasi akan memberikan berupa alert/pemberitahuan bahwa inputan tersebut invalid/salah.

1. **Boundary Value Analysis**

Teknik pengujian di mana kasus uji dirancang untuk memasukkan nilai-nilai pada batas. Pengujian positif pengujian berupa data input berada dalam batas nilai batas. Pengujian negatif adalah pengujian berupa data input berada di luar batas nilai batas. Analisis dari Nilai Batas didasarkan pada pengujian nilai batas partisi yang valid dan tidak valid. Perilaku di tepi partisi ekivalensi lebih cenderung salah daripada perilaku di dalam partisi valid, jadi batas adalah area di mana pengujian cenderung menghasilkan cacat.

Pengujian ini akan memeriksa nilai input di dekat batas yang memiliki peluang kesalahan lebih tinggi. Setiap partisi memiliki nilai maksimum dan minimum dan nilai maksimum dan minimum ini adalah nilai batas dari sebuah partisi. Adapun ilustrasi dari pengujian nilai batas dapat dilihat pada gambar di bawah ini.

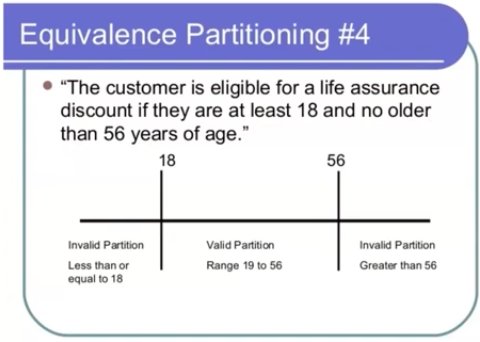


Dari ilustrasi tersebut dapat diketahui bahwa pengujian nilai batas adalah proses uji di batas antara *positive* *testing* dengan *negative testing*. Nilai batas untuk partisi yang *valid* adalah nilai batas yang *valid*. Oleh karena itu, pengujian dengan *Boundary Value Analysis* melakukan pengujian untuk setiap variabel mulai dari : nilai minimal, tepat di atas minimal, nilai nominal, tepat di bawah nilai maksimal, dan nilai maksimal.

Contoh dari pengujian nilai batas yaitu: Jika sebuah sistem hanya menerima nilai inputan untuk umur hanya dari 18 sampai 56. Dapat diketahui bahwa nilai minimal dari umur tersebut adalah 18, dan nilai maksimal dari umur adalah 56. Oleh karena itu pengujian batas seperti berikut.

|  |  |  |
| --- | --- | --- |
| ***Invalid***  (min-1) | ***Valid***  (min, min+1, nominal, max-1, dan max) | ***Invalid***  (max+1) |
| 17 | 18, 19, 37, 55, 56 | 57 |

***Valid Test Cases***: Pengujian yang *valid* dari nilai umur 18 sampai 56 terletak di atas nilai 17 dan di bawah nilai 57. Yaitu *valid* jika menginput nilai 18, 19, 37, 55, 56. ***Invalid Test Cases***: Ketika nilai di bawah nilai 18 dan di atas nilai 56 maka nilai akan invalid. Adapun ilustrasi dari permasalahan di atas sebagai berikut.



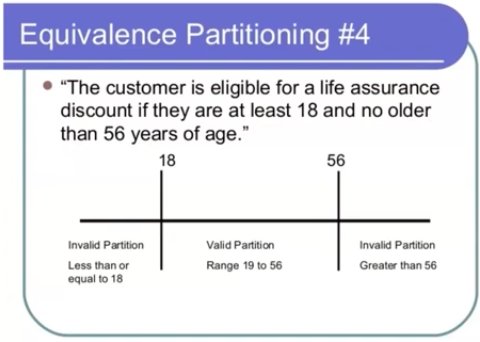
1. **Equivalence Partitioning**

Equivalence Partitioning juga dikenal sebagai Equivalence class partitioning (ECP). Ini adalah teknik pengujian perangkat lunak atau pengujian black box yang membagi domain input ke dalam kelas data, dan dengan bantuan kelas data ini, kasus uji dapat diturunkan. Kasus uji yang ideal mengidentifikasi kelas kesalahan yang mungkin memerlukan banyak kasus uji arbitrer(tidak tetap) untuk dieksekusi sebelum kesalahan umum diamati.

Dalam partisi kesetaraan, kelas kesetaraan dievaluasi untuk kondisi input yang diberikan. Setiap kali input diberikan, maka jenis kondisi input diperiksa, kemudian untuk kondisi input ini, kelas Ekuivalensi mewakili atau menggambarkan kumpulan status valid atau tidak valid. Adapun ilustrasi dari Equivalence Partitioning sebagai berikut.



Teknik pengujian yang membagi data input menjadi banyak partisi. Nilai dari setiap partisi harus diuji setidaknya sekali pengujian. Partisi dengan nilai yang *valid* digunakan untuk pengujian positif, dan partisi dengan nilai tidak *valid* digunakan untuk pengujian negatif. Contoh :



Gambar di atas sama seperti kasus pada *boundary value analysis*. Pengujian secara *equivalence partitioning* adalah pengujian yang akan membagi tiap-tiap nilai ke dalam kelompok masing. Contoh dari nilai kasus di atas yaitu minimal = 18 dan maksimal = 56. Maka terdapat 3 partisi yang dapat dibagi yaitu : 0-18, 19-56-, 56+. Contoh nilai inputan : 5, 40, dan 76 dapat diambil dari setiap bagian partisi untuk menguji setiap skenario yang ada.

# **BAB VII**

# **WEB ARCHITECTURE**

1. **Modern Web Architecture Explained**

Arsitektur pengujian adalah ilmu disiplin melihat aliran pengiriman dan mencari tahu apa, bagaimana, dan kapan pengujian untuk mencapai hasil terbaik. Pengujian ini menghasilkan kerangka kerja aktivitas pengujian yang kemudian dapat dicakup, disiapkan, dan dieksekusi.

Pengujian web arsitektur terbagi atas dua uji, yaitu secara Front-end dan Back-end. Pengujian secara Front-end meliputi uji aplikasi dari bagian user yang melakukan interaksi secara langsung. Contoh ilustrasi sebagai berikut.



Pada gambar di atas dapat diketahui bahwa user adalah pada bagian *Client*, dan *Server Side* adalah bagian yang tidak diketahui oleh user atau biasa disebut dengan *Back-end* aplikasi. Pada bagian *Server Side* user tidak akan mengetahui proses logika dari aplikasi, database, dan hal sensitif lainnya.

Web Architecture bekerja secara front-end dan back-end. Adapun proses yang terjadi selama user melakukan request untuk membuka sebuah web terbagi atas 2 proses, yaitu proses pada bagian user yang disebut front-end dan proses pada bagian yang tidak dapat dilihat oleh user disebut back-end.

**Frond-end :**

* Browser : User mengakses halaman web (HTTP/HTTPS Protocol)

**Back-end :**

* Back-End : Mengirim kembali sebagai HTML (Bahasa Standar untuk web browser)
* Browser : Menerjemahkan halaman web
* Browser: Download seluruh asset yang dibutuhkan seperti script, font, gambar Menghubungi pihak ketiga dari back-end seperti (metode pembayaran, analytics, maps)
* Browser : Menampilkan halaman web ke user

1. **Back-End Testing**

Pengujian back-end adalah pengujian yang dilakukan dari segi logika atau segi yang tidak dapat dilihat oleh user. Pengujian dari segi back-end biasanya melibatkan pengujian seperti Web Service Testing, Database Testing, dan Server Side Testing. Adapun analogi dari back-end dan front-end sebagai berikut.



Dari gambar tersebut dapat diketahui bahwa Back-end adalah sebuah hal yang dilakukan untuk membuat Front-end sesuai dengan yang diinginkan. Seperti pada gambar tersebut, developer ingin menampilkan singa mengaung, tentunya pada segi back-end atau belakang panggung menunjukkan bahwa singa tersebut diikat dan hanya menampilkan kepalanya saja.

1. **Front-End Testing**

Pengujian front-end adalah pengujian yang dilakukan dari segi tampilan yang dapat dilihat oleh user. Pengujian dari segi front-end biasanya melibatkan pengujian seperti Web Application Testing, Desktop Application Testing, dan Mobile Application Testing. Adapun ilustrasi dari pengujian front-end sebagai berikut.



Dari gambar tersebut dapat diketahui bahwa pengujian secara front-end akan menguji tampilan dari sebuah web, seperti jika user membuka web menggunakan perangkat mobile maka tampilannya dengan perangkat desktop tentunya tidaklah sama. Oleh karena itu, pengujian secara front-end pada web architecture sangat dibutuhkan agar website yang dibangun responsive terhadap segala jenis lingkungan/platform.

1. **How to Debug Front-End or Back-End Issues?**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

# **BAB VIII**

# **BUILD AND RELEASES**

1. **What is Software Build?**

Setelah mengembangkan modul perangkat lunak, pengembang mengubah kode sumber menjadi bentuk mandiri atau kode yang dapat dieksekusi. Kemudian tim pengembang menyerahkan hasil pengembangan kepada tim pengujian untuk melakukan pengujian. Kemudian tim penguji perangkat lunak akan memeriksa aplikasi tersebut. Jika terdapat beberapa bug dan jika tidak memenuhi persyaratan, maka tim pengujian perangkat lunak menolak pengembangan aplikasi tersebut. Build dilakukan sebelum proses perilisan perangkat lunak ke pasar.

Build adalah proses kompilasi, di mana seluruh source code file (text) diubah menjadi sebuah kode yang dapat dieksekusi. Adapun ilustrasi dari build dapat dilihat pada gambar di bawah ini.



Dari ilustrasi di atas, dapat dilihat bahwa proses sebelum merilis sebuah aplikasi ke publik/pasar harus melakukan kompilasi atau membuat sebuah kode teks yang dibuat menjadi dapat dieksekusi oleh platform tujuan aplikasi. Perbedaan utama dari build dan release terletak pada pengujian perangkat lunak. Aplikasi yang masih dalam tahap build wajib memerlukan pengujian mendalam, akan tetapi aplikasi yang telah masuk ke tahap release sudah tidak perlu melakukan pengujian apapun.

1. **What is Deployment?**

Pada tahapan pengembangan sistem, implementasi sistem merupakan tahapan akhir dalam pembangunan sistem. Pada tahapan implementasi, perpindahan dari sistem lama ke sistem baru dibagi menjadi 3 macam yaitu direct (secara langsung), parallel (setengah-setengah) dan juga phased (pergantian ke sistem baru dibuat per modul sistem).

Deployment dalam software dan web berarti mendorong perubahan atau pembaruan dari satu lingkungan penyebaran ke lingkungan penyebaran lainnya. Saat menyiapkan situs web, kita akan selalu memiliki situs web langsung (localhost), yang disebut lingkungan langsung atau lingkungan produksi. Deployment adalah kegiatan yang bertujuan untuk menyebarkan aplikasi yang telah dikerjakan oleh para pengembang. Penyebarannya dapat melalui beragam cara tergantung dari jenis aplikasinya.

Deployment Activites meliputi melakukan tes sistem dan stres, melakukan tes penerimaan, mengonversi data yang ada, membangun materi pelatihan/melakukan pelatihan, mengonfigurasi dan mengatur lingkungan produksi, dan menyebarkan solusi. Testing adalah aktivitas utama implementasi beserta penyebaran dan mencakup pengujian unit, uji integrasi, tes kegunaan, tes sistem/kinerja/stres, dan tes penerimaan.

Development plan program adalah trade-off (perdagangan) di antara sumber daya yang tersedia, waktu yang tersedia, dan keinginan untuk mendeteksi dan memperbaiki kesalahan sebelum penyebaran sistem dilakukan.

1. **What is Release?**

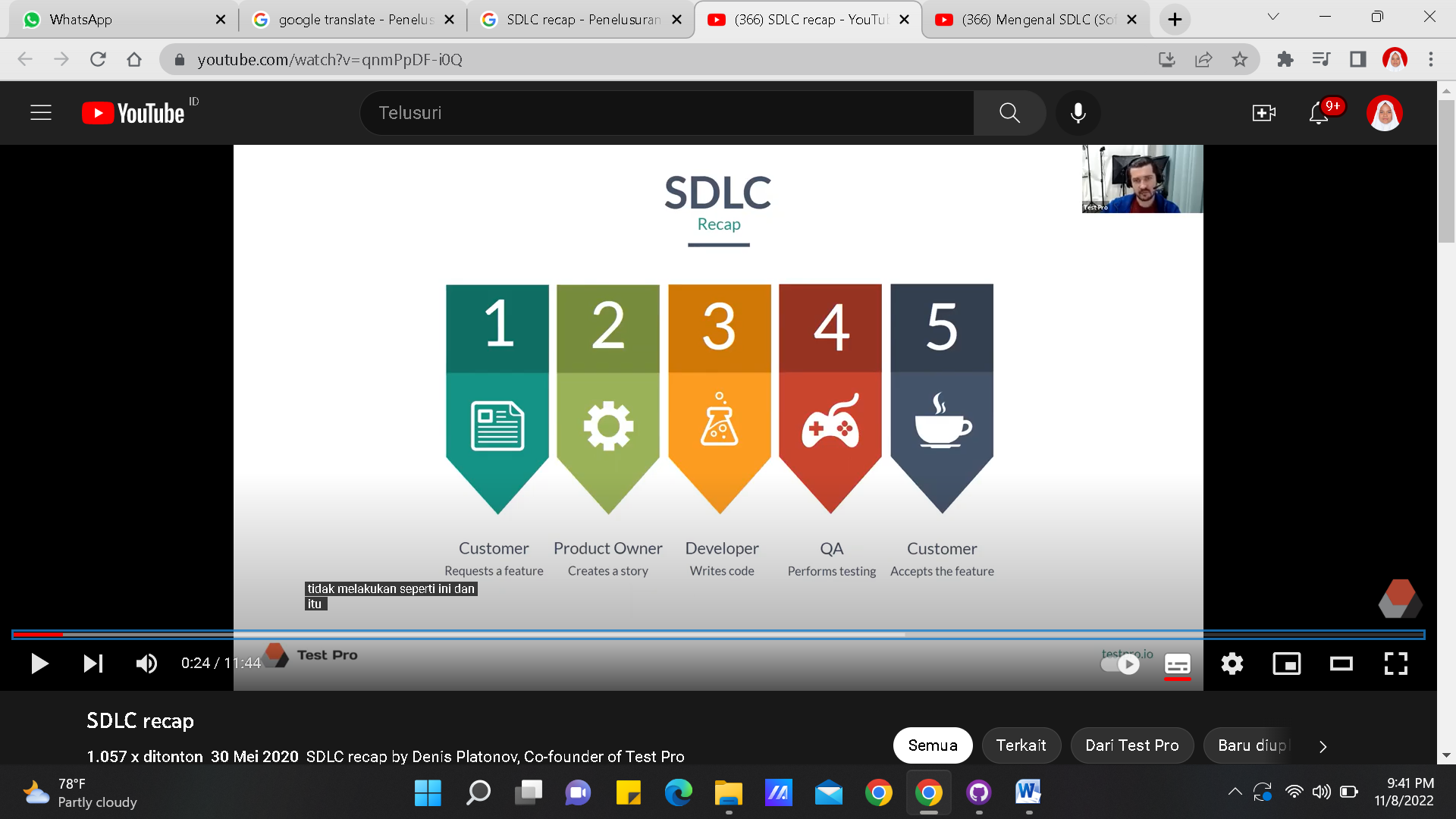
Rilis adalah aktivitas terakhir setelah menyelesaikan pengembangan dan pengujian. Setelah menguji pembuatan aplikasi, tim pengujian mengesahkan perangkat lunak itu dan mengirimkannya kepada pelanggan. Dimungkinkan untuk satu rilis memiliki beberapa build. Oleh karena itu, perangkat lunak yang dikirim ke pelanggan setelah menyelesaikan fase pengembangan dan pengujian. Selain itu, rilis ini didasarkan pada build, dan dapat memiliki beberapa build.

Build mengacu pada perangkat lunak mandiri yang dihasilkan setelah mengonversi kode sumber ke kode yang dapat dieksekusi dan dapat dijalankan di komputer. Rilis ini di sisi lain, adalah distribusi versi final suatu aplikasi. Dengan demikian, definisi-definisi ini menjelaskan perbedaan mendasar antara Build dan Rilis. Jadi perbedaan utama antara Build dan Rilis dalam Pengujian Perangkat Lunak yaitu :

* Build adalah versi perangkat lunak yang diserahkan oleh tim pengembangan ke tim pengujian untuk tujuan pengujian, Sedangkan
* Rilis adalah perangkat lunak yang tim uji berikan kepada pelanggan untuk dicoba

1. **SDLC Recapitulation**

Recapan SDLC adalah recap cepat pada siklus hidup pengembangan perangkat lunak. Disini saya akan menjelaskan siklus recapan SDLC melalui gambar dibawah ini :



Yang dimana beberapa pelanggan akan menulis ulasan tentang fitur fitur aplikasi yang kurang menarik atau fitur fitur yang susah digunakan sehingga pelanggan akan memberikan ulasan tentang aplikasi tersebut. Kemudian pemilik produk akan mencatat semua ulasan pelanggan tentang aplikasinya. Selanjutnya, pemilik produk akan menyerahkan cacatan ulasan pelanggan kepada pengembang yang dimana pengembang akan memperbaiki semua fitur fitur yang dianggap pelanggan kurang menarik. Kemudian pengambang akan memberikan fitur barunya kepada QA atau bisa disebut dengan tim penguji. Tim penguji akan melakukan pengujiannya terhadap fitur fitur baru. Setelah pengujian selesa maka pemilik produk siap menerima fitur baru dan pemilik produk akan memberikan kepada pelanggan.

1. **What is Sanity Testing?**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus.

Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum.

Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc,

1. **What is User Acceptance Testing?**

User Acceptance Testing (UAT) adalah jenis pengujian yang dilakukan oleh pengguna akhir atau klien untuk memverifikasi/menerima sistem perangkat lunak sebelum memindahkan aplikasi perangkat lunak ke lingkungan produksi. UAT dilakukan pada tahap akhir pengujian setelah pengujian fungsional, integrasi dan sistem dilakukan.

Tujuan utama UAT adalah untuk memvalidasi aliran bisnis ujung ke ujung. Itu tidak fokus pada kesalahan kosmetik, kesalahan ejaan atau pengujian sistem. Pengujian Penerimaan Pengguna dilakukan dalam lingkungan pengujian terpisah dengan pengaturan data seperti produksi. Ini adalah jenis pengujian kotak hitam di mana dua atau lebih pengguna akhir akan terlibat. UAT dilakukan oleh Klien dan Pengguna akhir.

Kebutuhan Pengujian Penerimaan Pengguna muncul setelah perangkat lunak telah menjalani pengujian Unit, Integrasi dan Sistem karena pengembang mungkin telah membangun perangkat lunak berdasarkan dokumen persyaratan dengan pemahaman mereka sendiri dan perubahan yang diperlukan lebih lanjut selama pengembangan mungkin tidak dikomunikasikan secara efektif kepada mereka, jadi untuk menguji apakah hasil akhir produk diterima oleh klien/pengguna akhir, pengujian penerimaan pengguna diperlukan. Berikut adalah kriteria entri untuk User Acceptance Testing:

1. Persyaratan Bisnis harus tersedia.
2. Kode Aplikasi harus dikembangkan sepenuhnya
3. Pengujian Unit, Pengujian Integrasi & Pengujian Sistem harus diselesaikan
4. Tidak Ada Showstoppers, Cacat Tinggi, Sedang dalam Fase Uji Integrasi Sistem
5. Hanya kesalahan kosmetik yang dapat diterima sebelum UAT
6. Pengujian Regresi harus diselesaikan tanpa cacat besar
7. Semua cacat yang dilaporkan harus diperbaiki dan diuji sebelum UAT
8. Matriks ketertelusuran untuk semua pengujian harus diselesaikan
9. Lingkungan UAT harus siap
10. Menandatangani surat atau komunikasi dari Tim Pengujian Sistem bahwa sistem siap untuk eksekusi UAT

# **BAB IX**

# **QUALITY ASSURANCE**

1. **Quality Assurance (QA)**

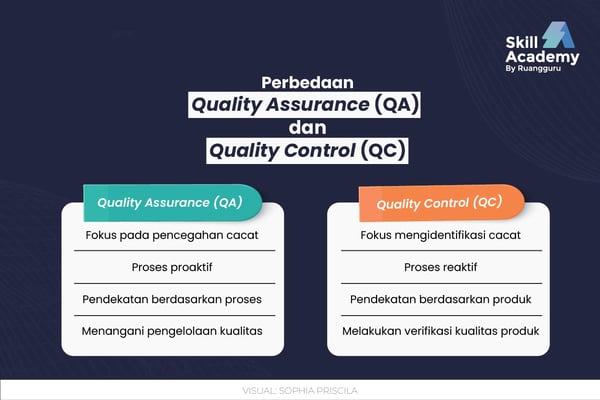
Quality Assurance adalah suatu profesi yang berperan untuk memastikan kualitas suatu produk sesuai dengan ketentuan yang berlaku dan menyiapkan segala kebutuhan dari aplikasi yang dibangun oleh perusahaan sehingga dapat bekerja dengan baik.. QA biasa ditemukan dalam perusahaan manufaktur dan perusahaan software.z

Quality Assurance Testing merupakan serangkaian proses sistematis guna menentukan apakah suatu produk dan jasa harus memenuhi syarat yang ditentukan. QA akan menentukan serta menetapkan persyaratan dalam membuat atau mengembangkan produk tertentu agar memiliki kualitas yang baik. Bukan tanpa alasan mengapa kualitas suatu produk sangat penting diperhatikan.

Kualitas terbaik dari adalah cara paling utama menjaga kredibilitas suatu perusahaan, selain itu juga cara meningkatkan kepercayaan konsumen, proses kerja hingga membuat perusahaan yang mampu membuat mereka bersaing dengan kompetitor. Quality Assurance artinya menggunakan pendekatan proses agar tidak memunculkan produk yang cacat.

Itulah mengapa biasanya QA juga akan melakukan monitoring pembuatan produk mulai dari tahap perencanaan hingga proses pengujian. Kegiatan itu dilakukan demi mengurangi proses pengulangan pembuatan atau rework, sehingga proses kerja menjadi lebih efisien serta dapat menghindari keluhan dari konsumen.

1. **Perbedaan dengan Quality Control (QC)**



Perbedaan Quality Control (QC) dan Quality Assurance (QA) terletak pada tugas dan tanggung jawab masing-masing. Walaupun sama-sama dalam satu departemen, QA memiliki peran dalam menjamin kualitas, sementara QC memiliki fungsi sebagai pengendali kualitas dari produk yang dihasilkan oleh perusahaan sesuai dengan standar keandalan, kegunaan, kinerja maupun standar lainnya.

QA memiliki tanggung jawab dalam memastikan sebuah produk sebelum dilepas ke pasaran, sebelum dirilis produk harus sudah memenuhi semua standar kualitas dalam setiap komponen. Pejabat staf QA memiliki kewajiban untuk aktif melakukan monitoring dan serangkaian pengujian dalam menetapkan kualitas pada pembeli.

Hal ini berbeda dengan QC dengan tanggung jawab memeriksa produk sebelum dan hingga setelah proses produksi menetapkan standar kualitas yang diperlukan. Pejabat QC memiliki hak menerima atau menolak produk yang akan dilepas ke pasaran, sehingga ketika ditemukan produk cacat maka akan dikembalikan ke bagian produksi.

Kedua posisi ini memiliki keterkaitan dan dapat bekerja secara bersamaan hingga saling berkolaborasi. Jenjang karir dari QA bisa mencapai tingkatan seorang Project Manager jika memiliki pengalaman dalam melakukan analisis dan melakukan audit suatu produk. Selain itu jenis pekerjaan ini juga bisa menjadikan DevOps, tugasnya mengotomatisasi proses tahap pengembangan aplikasi.

Untuk bisa menekuni pekerjaan ini seseorang bisa memulai karir lebih dulu menjadi Customer Experience Leader atau IT Management dengan peran pentingnya dalam memegang kendali penuh dari setiap proses pengembangan produk serta memiliki kewajiban mengutamakan kebutuhan dari konsumen atau pelanggan.

1. **Peran dan Tanggung Jawab Quality Assurance**

Pada umumnya tugas dari Quality Assurance adalah menjamin kualitas produk dari suatu perusahaan yang akan dijual atau masih dalam proses pengembangan. Selain itu QA juga memiliki beberapa tugas lain, di antaranya adalah sebagai berikut ini:

1. Membuat perencanaan terhadap pengujian dan kasus pengujian secara terperinci serta komprehensi terstruktur.
2. Melakukan tafsir, membangun dan mematuhi standar terhadap jaminan kualitas dari perusahaan yang menjual produk atau jasa.
3. Melakukan analisis terhadap keluhan konsumen dan ketidaksesuaian kualitas, selain itu juga mencari akar masalah serta tindakan penyelesaian yang sesuai dengan visi perusahaan.
4. Melakukan pengembangan standar baru dalam produksi sesuai dengan kebutuhan dan membuat protokol pengujian.
5. Melakukan dokumentasi aktivitas jaminan kualitas dalam bentuk laporan dan audit secara internal dalam perusahaan.
6. Memastikan produk yang dibuat sudah memenuhi standar perusahaan dan kebutuhan konsumen atau para pelanggan.
7. Melakukan dokumentasi perbaikan terhadap produk setelah dilakukan pengujian sebelum nantinya dijual ke pasaran.
8. Melakukan dokumentasi berupa catatan perbaikan yang dijadikan sebagai referensi terhadap produk setelah dilakukan pengujian.
9. Melakukan penyusunan terkait perencanaan Prosedur Operasi Standar (SOP) proses produksi terhadap produk dan layanan.
10. Bekerja sama, berkolaborasi dengan tim internal agar menemukan solusi dalam pemecahan masalah yang dihadapi saat itu.
11. **Education Requirements for Quality Assurances**

Menjadi seseorang dari QA harus memahami skill dan kemampuan dasar untuk bisa bekerja secara profesional dan sesuai dengan kualifikasi kerja di suatu perusahaan, berikut di antara dari kemampuan yang dibutuhkan:

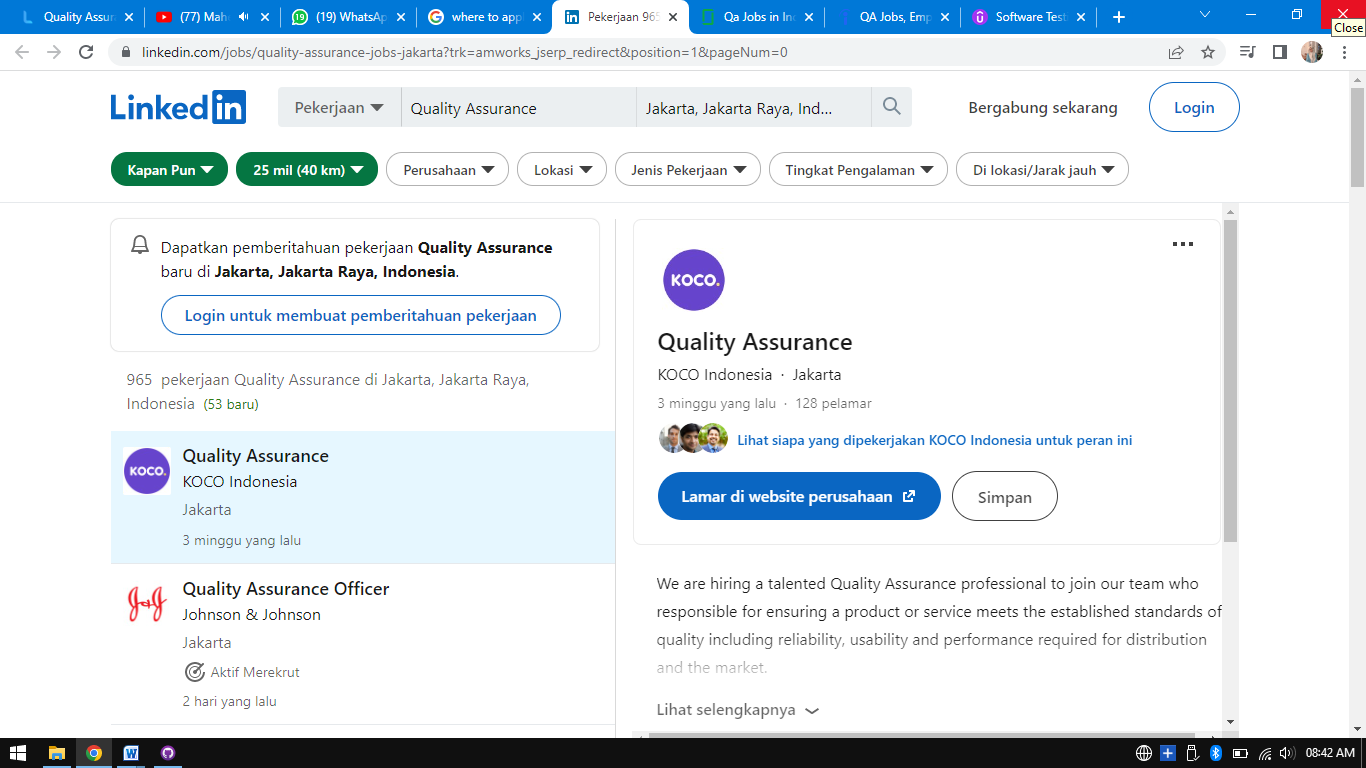
1. Memahami, aktif berkomunikasi dan membaca data dengan sangat baik.
2. Memiliki keahlian dalam manajemen proyek yang optimal.
3. Memahami sebuah produk yang bisa mempengaruhi pola pikir, kebiasaan dan kehidupan manusia.
4. Kemampuan berpikir kritis dan kemampuan analisis serta riset.
5. Keahlian dalam manajerial hingga penguasaan bahasa asing yang diperlukan oleh perusahaan.
6. Kemampuan analisa dan pengujian fungsional serta perbaikan dalam proses yang dilakukan.
7. Memiliki pola pikir pengujian, pengujian keamanan, otomasi hingga UAT (User Acceptance Testing).

Dalam menjalankan proses atau tugasnya, seorang QA membutuhkan dukungan peralatan atau tools yang kompatibel sesuai dengan pekerjaannya. Seperti salah satunya instalasi aplikasi GIT, selain harus memahami algoritma dan struktur data bahasa pemrograman yang dipakai dalam perusahaan tempat mereka bekerja. Keahlian untuk menjadi QA biasanya mirip, kurang lebih seperti yang disebutkan dan untuk hard skill biasanya disesuaikan dengan posisi yang dilamar. Misalnya Software Quality Assurance atau QA Engineer, maka harus bisa menguasai SDLC (Software Development Life Cycle), bahasa pemrograman seperti (Java, JavaScript, Python, C++ dll), testing tool, manual testing hingga automation testing.

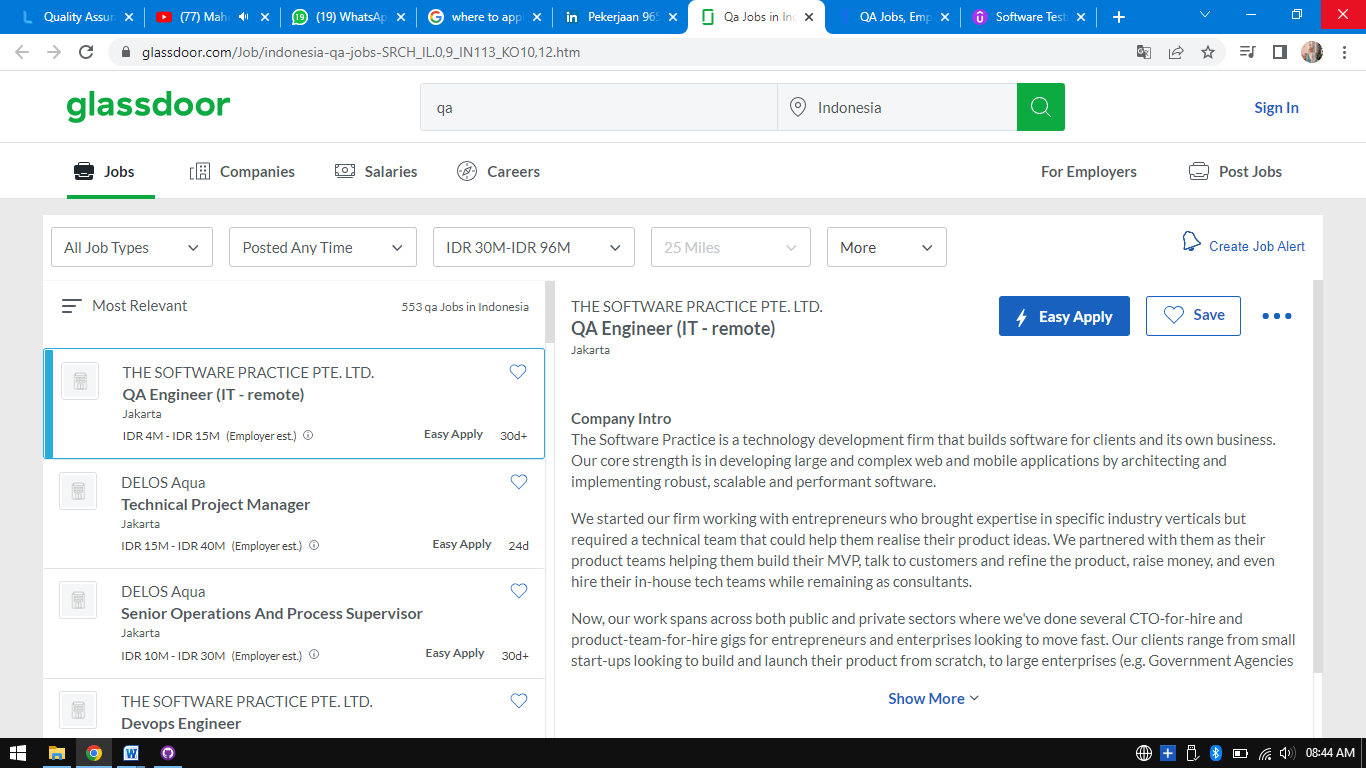
1. **Quality Assurance Jobs**

Saat ini, kebutuhan perusahaan terkait profesi Quality Assurance semakin meningkat. Oleh karena itu, lowongan kerja quality assurance banyak dibuka dan pastinya peluang untuk berpartisipasi pada bidang ini sangatlah besar. Banyak platform yang dapat digunakan untuk melamar pekerjaan sebagai Quality Assurance, diantaranya LinkedIn, Glassdoor, Indeed, dan Monster.

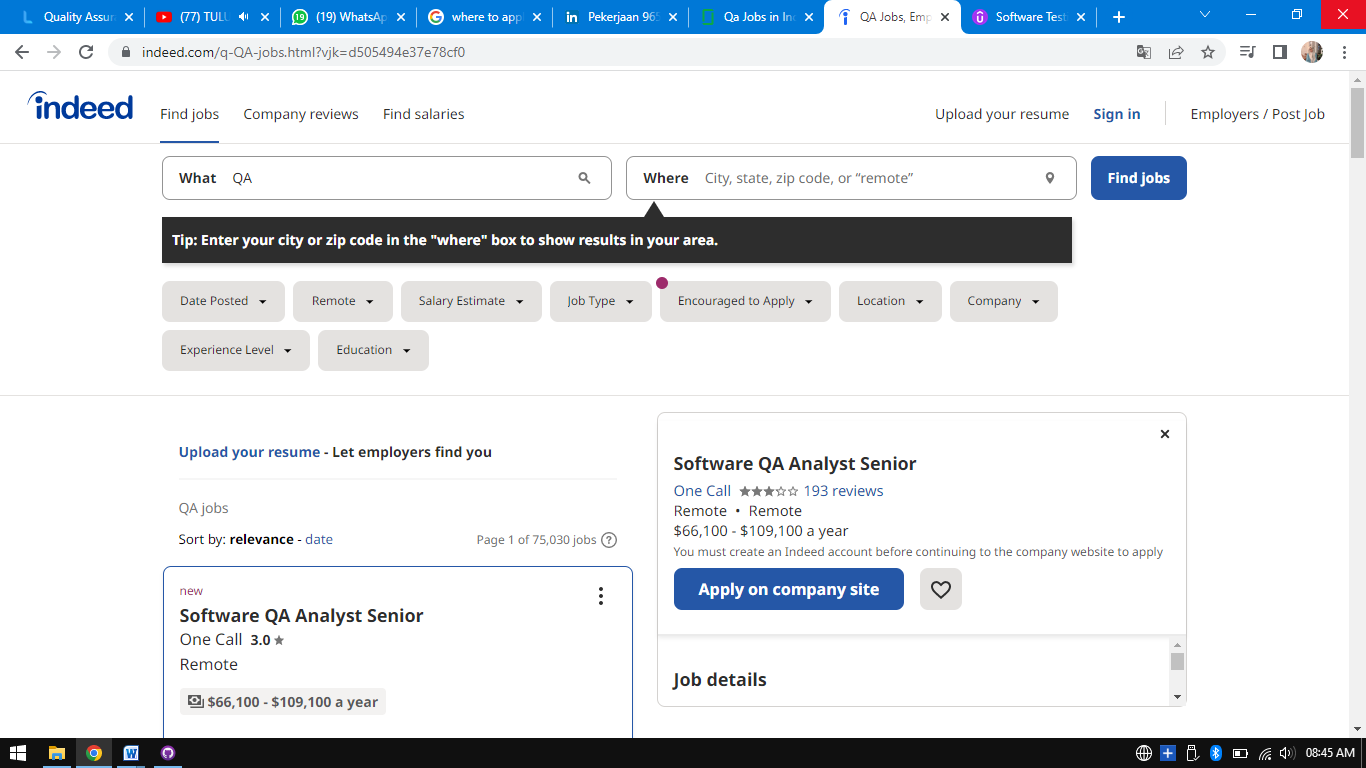
1. LinkedIn



1. Glassdoor



1. Indeed



1. Monster

