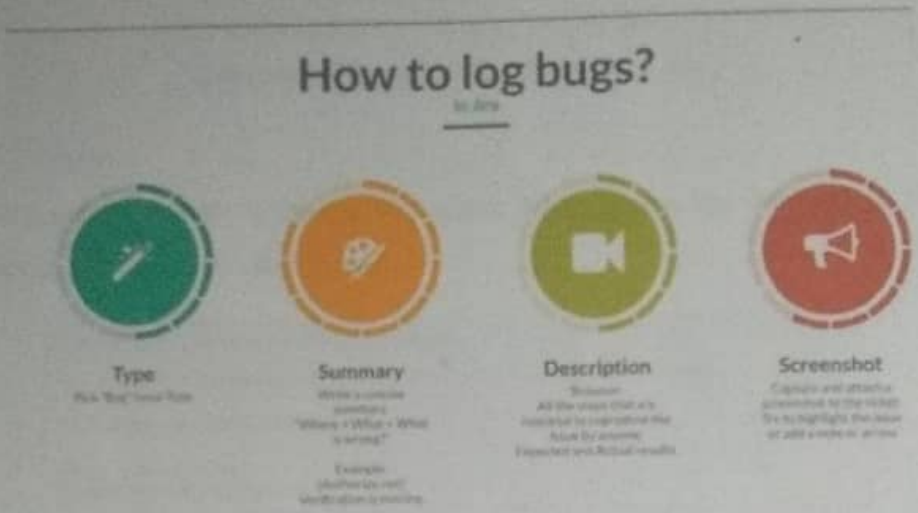
BAB III
BUGS

A. How to Report Bugs

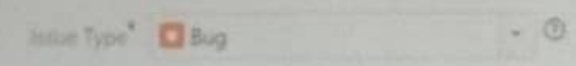
Untuk melaporkan *bug* dapat menggunakan Jira, berikut 4 langkah utama *log bugs* dalam Jira:



Gambar 3.1 *Log bugs* pada Jira

1. *Type*

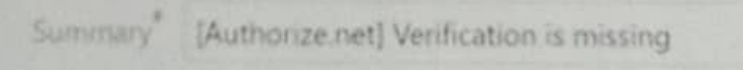
Pilih jenis masalah *bug*, yang mana tergantung pada proyek yang telah disiapkan di JIRA. Jenis masalah dapat berisi nilai berikut: *bug, story, epic, task* dan sebagainya untuk melaporkan kerusakan.



Gambar 3.2 Tampilan *type* pada Jira

2. *Summary*

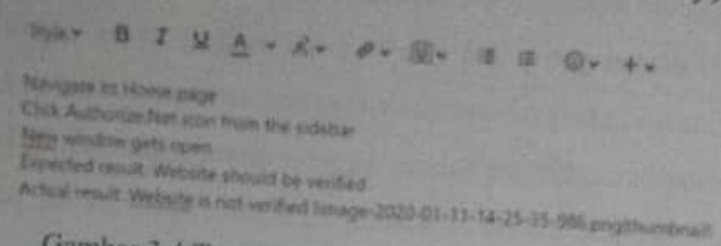
Menulis ringkasan singkat tentang apa dan dimana yang salah. Pesan yang dapat ditulis di *summary* yaitu [Authorize.net] Verification is missing.



Gambar 3.3 Tampilan *summary* pada Jira

3. *Description*

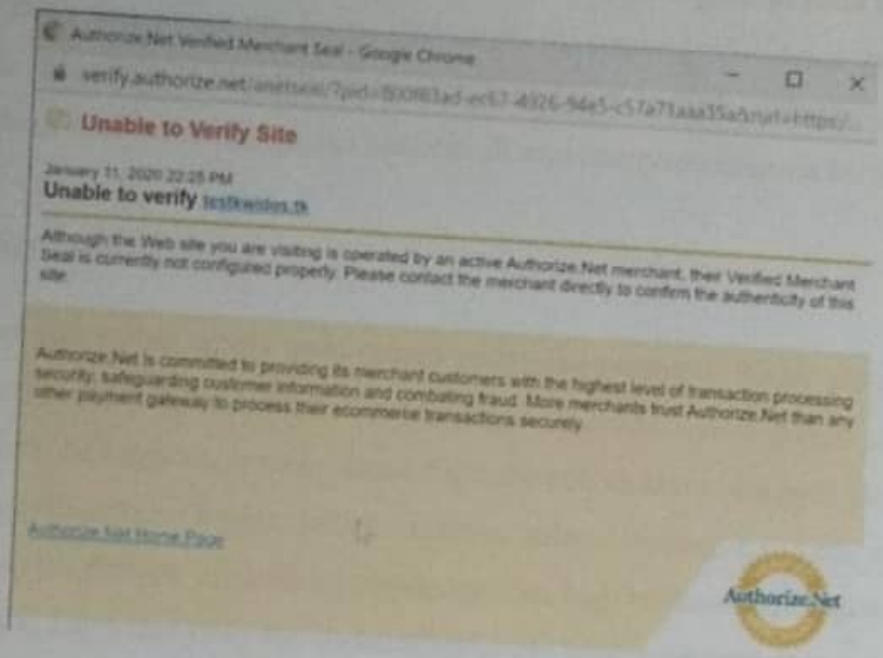
Deskripsi ini adalah bidang yang paling penting ketika membuat laporan *bug*, yang mana deskripsi ini sedikit lebih deskriptif dari ~~dari~~ ringkasan (*summary*).



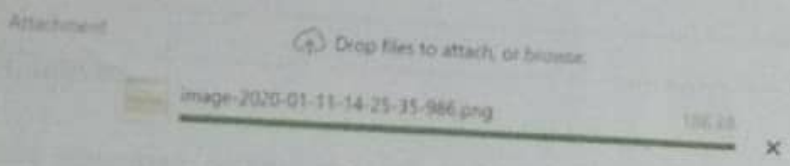
Gambar 3.4 Tampilan *description* pada Jira

4. *Screenshot*

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



Gambar 3.5 Tampilan jendela Authorize.Net pada Jira



Gambar 3.6 Tampilan *attachment* hasil screenshot pada Jira

BAB V
CROSS-PLATFORM AND CROSS-BROWSER TESTING

Cross-platform testing mengharuskan aplikasi berjalan ~~sama baiknya~~ berjalan dengan baik (sebaiknya sempurna) di semua jenis komputer, perangkat seluler dan sistem operasi seperti Windows, OS, Android, Macintosh, Chrome OS, dan lain-lain. *Cross-platform testing* menentukan semua kemungkinan masalah *caching* dan perilaku aplikasi seluler atau web pada berbagai perangkat yang berbeda.

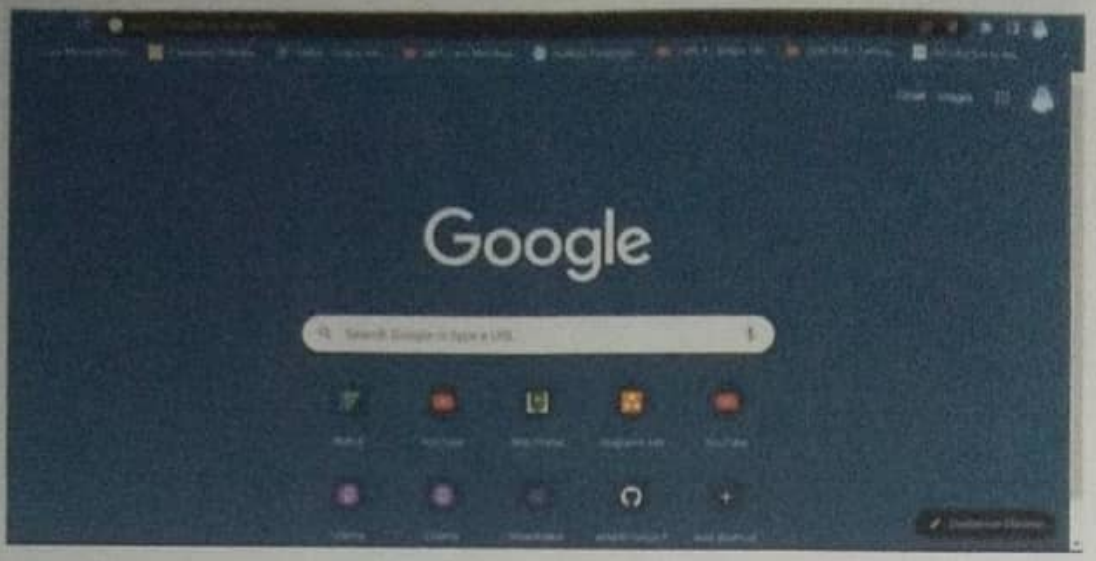
A. How to Test on Mobile Devices?

Untuk melakukan pengujian dengan menggunakan perangkat seluler secara langsung tentunya dibutuhkan sebuah perangkat seluler (*smartphone*). Namun, jika pengujian secara langsung ingin dilakukan pada perangkat partikular akan sedikit menyusahkan karena keterbatasan perangkat yang dimiliki. Solusi dari permasalahan tersebut adalah menggunakan simulator atau emulator pengguna, salah satunya melalui *browser* Chrome. Selain emulator, juga terdapat layanan *cloud* khusus, 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.

B. 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. Google Chrome menyediakan sarana untuk membuka *website mobile* di komputer tanpa harus menginstal ekstensi tambahan. Berikut cara menggunakannya:

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



Gambar 5.1 Halaman utama pada *browser* Chrome

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

Gambar 5.2 Menu inspeksi pada *browser* Chrome

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

A. 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 poin untuk diproduksi. Dengan kata lain *test plan* dapat disebut sebagai perencanaan atau skenario untuk melakukan *testing* yang akan dilakukan baik oleh *expert* atau pengguna 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 selalu 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 diuji. 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.

2. *Introduction*

   Pada bagian *introduction* dibuat untuk menjelaskan secara narasi, mengenai *testing* yang akan dilakukan terhadap suatu objek. 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*.

3. *Test Items*

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

4. *Features to be tested*

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

5. *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 uji pada saat *testing*.

6. *Approach/Test Strategy*

Bagian *Approach* adalah bagian yang digunakan untuk memberi deskirpsi mengenai cara yang dilakukan untuk melaksanakan *testing* dan disertakan dengan penjelasan mengenai cara yang digunakan.

7. *Item Pass/Fail Criteria*

Berisi tentang kriteria-kriteria yang harus dipenuhi sebelum berlanjut ke fase berikutnya. Kriteria yang dimaksud merangkup (mencakup) kriteria yang benar serta kriteria yang salah.

8. *Suspension Criteria*

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

9. *Test Deliverables*

Merupakan daftar dokumen-dokumen yang akan dihasilkan setelah *testing* selesai dilakukan.

10. *Testing Task*

Menjelaskan kegiatan *testing* beserta dengan pihak yang akan melaksanakan kegiatan tersebut.

B. 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~~ jenis 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 tes 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.

C. 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 adalah perangkat lunak yang telah menyelesaikan fase pengembangan dan pengujian.

Build mengacu pada perangkat lunak mandiri yang dihasilkan setelah mengonversi kode sumber ke kode yang dapat dieksekusi dan dapat dijalankan di komputer. Rilis juga disebut sebagai 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

D. What is Sanity Testing?

Sanity testing adalah sejenis pengujian perangkat lunak yang dilakukan setelah menerima pembuatan perangkat lunak, dengan sedikit perubahan dalam kode, atau fungsionalitas, untuk memastikan bahwa *bug* telah diperbaiki dan tidak ada masalah lebih lanjut yang ditimbulkan karena perubahan ini. Tujuannya adalah untuk menentukan bahwa fungsionalitas yang diusulkan bekerja seperti yang diharapkan. Jika *sanity testing* gagal, *build* akan ditolak untuk menghemat waktu dan biaya yang terlibat dalam pengujian yang lebih ketat.

Tujuannya adalah "bukan" untuk memverifikasi secara menyeluruh fungsionalitas baru tetapi untuk menentukan bahwa pengembang telah menerapkan beberapa rasionalitas saat memproduksi perangkat lunak. Contoh *sanity testing*:

Dalam proyek *e-commerce*, modul utama adalah halaman *login*, halaman beranda, halaman profil pengguna, pendaftaran pengguna, dll. Terdapat kecacatan proses pada halaman *login* ketika kata sandi menerima kurang dari empat karakter dan persyaratan menyebutkan bahwa kata sandi ini bidang tidak boleh di bawah delapan karakter. Oleh karena itu, cacat dilaporkan oleh tim pengujian ke tim pengembangan untuk menyelesaikannya. Kemudian tim pengembangan memperbaiki cacat yang dilaporkan dan mengirimkannya ke tim pengujian untuk dibersihkan. Kemudian tim penguji memeriksa apakah perubahan yang dilakukan berfungsi dengan baik atau tidak. Itu juga ditentukan apakah itu berdampak pada fungsi terkait lainnya. Sekarang ada fungsi untuk memperbarui kata sandi di halaman profil pengguna.

Keuntungan dari *sanity testing*:

1. *Sanity testing* membantu dengan cepat mengidentifikasi cacat pada fungsionalitas inti.
2. Dapat dilakukan dalam waktu yang lebih singkat karena tidak ada dokumentasi yang diperlukan untuk *sanity testing*.
3. Jika cacat ditemukan selama *sanity testing*, proyek akan ditolak untuk menghemat waktu pelaksanaan uji regresi.
4. Teknik pengujian ini tidak begitu mahal jika dibandingkan dengan jenis pengujian lainnya.