3 Alasan Dunia Tidak Butuh Software Testing

Tugas

Matakuliah : Kapita Sekekta

Dosen Pengampu : Roni Andarsyah, ST., M.KOM.,SFPC`

Oleh :

FAHIRA (1204044)
No. Telepon : 085262774355
Email : fahira1678@gmail.com

Universitas Logistik & Bisnis Internasional

PROGRAM STUDI DIPLOMA IV TEKNIK INFORMATIKA

UNIVERSITAS LOGISTIK DAN BISNIS INTERNASIONAL

BANDUNG

2023

Pemahaman Prinsip Testing, Functional testing, Non Functional testing, Structural testing dan Testing related to Change

1. Prinsip Testing

   Prinsip Testing adalah serangkaian pedoman untuk melakukan tes secara efektif dan efisien. Ada tujuh prinsip testing yang didefinisikan oleh ISTQB (International Software Testing Qualifications Board) yang mencakup:

   1. Pengujian menunjukkan keberadaan bug: Prinsip ini menyatakan bahwa pengujian perangkat lunak dirancang untuk menemukan kesalahan atau bug dalam perangkat lunak.
   2. Pengujian lengkap tidak mungkin dilakukan: Prinsip ini menyatakan bahwa tidak mungkin untuk melakukan pengujian yang lengkap pada perangkat lunak.
   3. Early Testing: Prinsip ini menyatakan bahwa pengujian perangkat lunak harus dimulai sejak awal dalam siklus pengembangan perangkat lunak.
   4. Defect clustering: Prinsip ini menyatakan bahwa sebagian besar kesalahan dalam perangkat lunak terjadi pada area tertentu atau "cluster" dalam perangkat lunak.
   5. Pesticide Paradox: Prinsip ini menyatakan bahwa jika pengujian dilakukan dengan pengujian yang sama berulang-ulang, maka pengujian tersebut tidak efektif dalam menemukan bug yang baru.
   6. Testing is context dependent: Prinsip ini menyatakan bahwa pengujian perangkat lunak harus disesuaikan dengan konteks perangkat lunak dan kebutuhan pengguna.
   7. Absence-of-errors fallacy: Prinsip ini menyatakan bahwa menemukan sedikit atau tidak ada kesalahan dalam pengujian tidak menjamin bahwa perangkat lunak bebas dari kesalahan.

2. Functional Testing

   Functional testing adalah jenis pengujian yang dilakukan untuk memeriksa apakah fungsi-fungsi perangkat lunak berjalan dengan benar sesuai dengan spesifikasi dan persyaratan. Contoh dari pengujian fungsional meliputi pengujian unit, pengujian integrasi, pengujian sistem, dan pengujian penerimaan.

3. Non-Functional Testing

Non-functional testing adalah jenis pengujian yang dilakukan untuk memeriksa karakteristik-karakteristik non-fungsional dari perangkat lunak, seperti kinerja, keamanan, keterandalan, dan skalabilitas. Contoh dari pengujian non-fungsional meliputi pengujian beban, pengujian stres, pengujian keamanan, dan pengujian ketersediaan.

4. Structural Testing

Structural Testing adalah jenis pengujian perangkat lunak yang dilakukan untuk memastikan bahwa struktur internal kode perangkat lunak memenuhi persyaratan dan spesifikasi yang telah ditentukan. Tujuan utama dari structural testing adalah untuk memastikan bahwa setiap bagian dari kode perangkat lunak bekerja dengan benar dan tidak memiliki cacat atau kelemahan.

Structural testing melibatkan pengujian unit, di mana setiap unit kode diuji secara terpisah, dan pengujian integrasi, di mana berbagai unit kode diuji dalam konteks sistem yang lebih besar. Ada beberapa teknik yang digunakan dalam structural testing, seperti coverage testing, path testing, dan mutation testing.

Coverage testing adalah teknik yang digunakan untuk mengukur sejauh mana kode diuji selama pengujian. Path testing, di sisi lain, adalah teknik yang digunakan untuk memastikan bahwa setiap jalur kode telah diuji secara terpisah. Sedangkan mutation testing adalah teknik yang digunakan untuk menemukan bug atau kelemahan dalam kode dengan mengubah kode asli dan memeriksa apakah pengujian masih berhasil atau gagal.

Keuntungan dari structural testing adalah bahwa ini memastikan bahwa kode perangkat lunak bekerja dengan benar dan memenuhi persyaratan dan spesifikasi yang telah ditentukan. Hal ini dapat membantu mencegah bug dan kelemahan pada perangkat lunak, serta mempercepat waktu pengembangan perangkat lunak dengan mengidentifikasi masalah dengan cepat.

Namun, structural testing juga memiliki beberapa keterbatasan, seperti sulitnya melakukan pengujian untuk semua kemungkinan jalur kode dan ketergantungan pada kode asli yang dapat menyebabkan kesalahan dalam pengujian. Oleh karena itu,

structural testing harus dilakukan dengan hati-hati dan dengan menggunakan teknik pengujian yang tepat untuk memastikan kualitas perangkat lunak yang baik.

5. Testing Related to Change

   Testing Related to Change atau pengujian yang berkaitan dengan perubahan adalah jenis pengujian yang dilakukan ketika ada perubahan yang dilakukan pada perangkat lunak. Perubahan bisa berasal dari perbaikan bug, perubahan fungsional, perubahan desain, atau perubahan konfigurasi. Tujuan dari pengujian yang berkaitan dengan perubahan adalah untuk memastikan bahwa perubahan tersebut tidak mempengaruhi fungsionalitas atau kinerja perangkat lunak secara negatif. Pengujian yang berkaitan dengan perubahan meliputi pengujian regresi, di mana pengujian dilakukan untuk memastikan bahwa perubahan yang dilakukan tidak mempengaruhi fungsi yang telah diperiksa sebelumnya. Juga termasuk dalam pengujian ini adalah pengujian integrasi, di mana perubahan diuji dalam konteks sistem yang lebih besar, dan pengujian performa, di mana perubahan diuji dalam kondisi beban yang berbeda. Pentingnya pengujian yang berkaitan dengan perubahan adalah untuk memastikan bahwa perangkat lunak tetap berfungsi dengan baik dan memenuhi persyaratan dan spesifikasi yang ada setelah perubahan dilakukan. Jika perubahan tidak diuji dengan baik, maka dapat menyebabkan kerusakan pada sistem secara keseluruhan dan mengakibatkan kerugian pada pengguna. Oleh karena itu, pengujian yang berkaitan dengan perubahan harus dilakukan dengan cermat dan terus-menerus untuk memastikan kualitas perangkat lunak yang baik.

Contoh Functional testing, Non Functional testing, Structural testing dan Testing related to Change

1. Contoh Functional testing
   - Pengujian fungsional pada sebuah aplikasi e-commerce untuk memeriksa apakah fitur seperti menambahkan item ke keranjang, checkout, dan membayar bekerja dengan benar sesuai persyaratan.
   - Pengujian fungsional pada aplikasi perbankan untuk memeriksa apakah fitur seperti transfer dana, cek saldo, dan pembayaran tagihan bekerja dengan benar sesuai persyaratan.

2. Contoh Non Functional testing
    - Pengujian keamanan pada sebuah aplikasi web untuk memeriksa apakah data pengguna terlindungi dengan baik dan tidak dapat diakses oleh pihak yang tidak berwenang.
    - Pengujian performa pada sebuah aplikasi game untuk memeriksa apakah game berjalan lancar tanpa lag atau jeda yang terlalu lama.
    - Pengujian usability pada sebuah aplikasi mobile untuk memeriksa apakah antarmuka pengguna mudah digunakan dan dapat diakses oleh pengguna dengan berbagai tingkat pengalaman.

3. Contoh Structural testing
    - Pengujian unit pada sebuah aplikasi untuk memeriksa apakah setiap unit kode bekerja dengan benar dan tidak memiliki bug atau kelemahan.
    - Pengujian integrasi pada sebuah sistem untuk memeriksa apakah berbagai unit kode dapat bekerja bersama secara efektif dan efisien.
4. Contoh Testing related to Change
    - Pengujian regresi pada sebuah aplikasi setelah dilakukan perbaikan bug untuk memastikan bahwa perbaikan tersebut tidak mempengaruhi fungsi yang telah diperiksa sebelumnya.
    - Pengujian integrasi pada sebuah sistem setelah dilakukan perubahan fungsional untuk memastikan bahwa perubahan tersebut tidak mempengaruhi kinerja sistem secara negatif.

Intisari dari Pelatihan

Kegagalan proyek perangkat lunak dapat terjadi dalam beberapa bentuk, antara lain:

- Proyek dibatalkan sebelum selesai.
- Proyek selesai, tetapi tidak pernah digunakan.
- Perangkat lunak tidak memberikan manfaat bagi pengguna.
- Perangkat lunak tidak memenuhi keinginan pengguna.

Stages of Testing

1. Unit testing

   Tipe unit testing : Black Box testing dan White Box testing.

2. Integration testing

   Tipe integration testing : User interface testing, use-case testing, interaction testing, dan system interface testing.
3. System testing

   Tipe system testing : Requirements testing, usability testing, security testing, performance testing, dan documentation testing.
4. Acceptance testing

   Tipe Acceptance testing : alpha testing dan beta testing

3 Alasan Dunia Tidak Butuh Software Testing :

1. Waktu dan Biaya Pengembangan Menjadi Lebih Lama
2. Keyakinan Berlebihan pada pengalaman proyek sebelumnya
3. Keterbatasan sumber daya yang dimiliki.