PROSES TESTING: (BLACK BOX TESTING, WHITE BOX TESTING) DALAM PERANCANGAN PERANGKAT LUNAK

Mata Kuliah: Software Engineering

DOSEN: Yudhi Fajar Saputra, S.Kom., M.Sc

Pertemuan ke-10

Topik Bahasan ke-

SEMESTER : 3/ TA. 2024-2025

KODE MK/SKS: MKP001/3 SKS

PRODI INFORMATIKA/ILMU KOMPUTER

UNIVERSITAS WIDYA GAMA MAHAKAM SAMARINDA

Nama Mata Kuliah	: Software Engineeing/Rekayasa Perangkat Lunak
Kode Mata Kuliah/SKS	: **MKP _____/3 SKS**
Dosen	: **Yudhi Fajar Saputra,**
Semester	: **3/ 2024**
Hari Pertemuan / Jam	: -
Tempat Pertemuan	: **Ruang Kelas A.06**

Dalam dunia pengembangan perangkat lunak, kualitas dan keandalan menjadi dua aspek penting yang harus dijaga. Pengujian perangkat lunak (software testing) adalah salah satu tahap untuk memastikan bahwa aplikasi berfungsi sesuai kebutuhan dan bebas dari kesalahan yang mungkin mengganggu pengguna. Untuk mencapai kualitas yang optimal, ada dua pendekatan utama dalam pengujian perangkat lunak, yaitu black box testing dan white box testing.

Black box testing menitikberatkan pada pengujian fungsional aplikasi tanpa melihat ke dalam struktur atau kode internalnya. Dengan pendekatan ini, tester memperlakukan perangkat lunak sebagai "kotak hitam" dan hanya fokus pada input serta output yang dihasilkan, memastikan bahwa aplikasi memenuhi spesifikasi dan harapan pengguna.

Di sisi lain, white box testing adalah pendekatan yang lebih detail, di mana penguji memiliki akses ke kode sumber dan memeriksa alur logika serta struktur di dalam aplikasi. Melalui pengujian ini, penguji dapat memastikan bahwa setiap jalur kode telah diuji, setiap cabang logika berfungsi dengan baik, dan tidak ada kesalahan tersembunyi di dalam sistem.

Kedua metode ini, memiliki fokus yang berbeda, dimana black box testing memastikan fungsi dari sudut pandang pengguna akhir, sedangkan white box testing memberikan jaminan kualitas pada level kode. Dengan kombinasi kedua pendekatan ini, perangkat lunak dapat diuji secara komprehensif, menjamin fungsionalitas sekaligus integritas kode.

1. BLACK BOX TESTING

Black Box Testing adalah metode pengujian perangkat lunak yang menguji fungsionalitas aplikasi atau sistem tanpa mempertimbangkan bagaimana kode atau struktur internalnya bekerja [1]. Black box testing seringkali digunakan untuk memverifikasi bahwa perangkat lunak bekerja sesuai dengan spesifikasi atau kebutuhan fungsional, tanpa harus memahami kode sumbernya.

1) Konsep Black Box Testing

   beberapa konsep black box testing adalah

   a) Berbasis Fungsionalitas: Black box testing hanya berfokus pada output yang

dihasilkan oleh perangkat lunak ketika diberi input tertentu. Penguji melihat apakah output tersebut sesuai dengan spesifikasi atau persyaratan pengguna. selain itu Pengujian ini memeriksa bagaimana aplikasi berfungsi sesuai kebutuhan, bukan bagaimana aplikasi tersebut dikembangkan atau dibangun.

b) Tidak Memerlukan Akses Source Code: Dalam black box testing, penguji tidak perlu memiliki pemahaman atau akses terhadap Source Code. Tester atau penguji menganggap perangkat lunak sebagai "kotak hitam" dan hanya menganalisis berdasarkan masukan (input) dan keluaran (output) yang terlihat. Sehingga membuat pengujian secara netral terhadap implementasi dan desain teknis, hanya berfokus pada hasil.

c) Fokus pada Pengguna Akhir: Karena pendekatan ini tidak bergantung pada pemahaman struktur internal, penguji dapat mengevaluasi perangkat lunak dengan sudut pandang pengguna akhir. Pengujian dilakukan untuk memastikan bahwa aplikasi memberikan penggunaan yang baik dan hasil yang diharapkan oleh pengguna akhir

2) Teknik-teknik Black Box Testing

beberapa teknik black box testing adalah:

a) **Equivalence Partitioning:** Teknik **Equivalence Partitioning** membagi data uji ke dalam beberapa kelas yang sama atau mirip, dengan asumsi bahwa data dalam satu kelas akan memberikan hasil yang sama. Pengujian hanya dilakukan pada satu atau beberapa data dari setiap kelas [1], yang mengurangi jumlah pengujian yang diperlukan namun tetap mempertahankan efektivitas

b) Boundary Value Analysis: Teknik Boundary Value Analysis fokus pada pengujian batas dari data input, seperti nilai minimum dan maksimum. Batas data sering menjadi sumber kesalahan, sehingga menguji di titik-titik batas dapat membantu menemukan kesalahan yang mungkin terjadi [2].

c) Decision Table Testing: Teknik Decision table testing menggunakan tabel keputusan yang menggambarkan berbagai kombinasi kondisi dan tindakan atau output yang diharapkan. Teknik ini digunakan untuk menangani situasi kompleks dimana berbagai kombinasi kondisi menghasilkan berbagai hasil [3].

d) State Transition Testing: Teknik State Transition Testing menguji transisi antar keadaan dalam suatu sistem. Pengujian ini memeriksa bagaimana sistem bereaksi ketika berpindah dari satu kondisi atau status ke status lain berdasarkan input tertentu [2].

e) Use Case Testing: Teknik Use Case Testing menguji perangkat lunak

berdasarkan skenario-skenario penggunaan nyata dari sudut pandang pengguna. Pengujian ini dilakukan dengan mengikuti alur proses yang umum ditemui oleh pengguna untuk memastikan perangkat lunak berjalan sesuai dengan kebutuhan di situasi nyata.

3) Contoh Black Box Testing

Sebagai contoh, untuk aplikasi perbankan, tester mungkin menguji fungsionalitas seperti login, transfer dana, dan pembayaran tagihan. Black box testing dapat menguji apakah:

a) Pengguna dapat masuk dengan kredensial yang benar
b) Sistem menampilkan pesan kesalahan yang benar untuk input yang salah
c) Transfer dana berhasil dengan jumlah yang benar dan saldo diperbarui sesuai transaksi

2. WHITE BOX TESTING

White Box Testing adalah metode pengujian perangkat lunak yang berfokus pada pengujian struktur internal dan logika dari kode program. Dalam pengujian ini, penguji memiliki akses ke source code dan memahami alur logika, struktur, dan detail internal lainnya. White box testing sering disebut sebagai **clear box testing** atau **glass box testing** karena pengujian ini dibutuhkan pengetahuan terhadap sistem di balik layar.

1) Konsep White Box Testing

beberapa konsep white box testing adalah

a) Berbasis Source Code : Menemukan dan memperbaiki kesalahan dalam logika program dan Source Code
b) Mengoptimalkan performa kode program
c) Fokus pada Cakupan Kode.

2) Teknik White Box Testing

beberapa teknik white box testing adalah

a) **Statement Coverage**: Teknik ini mengukur persentase pernyataan (statement) dalam kode yang telah dieksekusi setidaknya satu kali selama pengujian, sehingga bisa mendeteksi kode yang tidak terpakai atau redundan [1]
b) **Branch Coverage**: Branch coverage, atau cakupan cabang artinya memastikan bahwa setiap cabang dari struktur pengkondisian (if-else, switch) diuji [2].
c) **Path Coverage**: Path coverage mencakup setiap jalur yang mungkin dilalui oleh eksekusi program, baik jalur utama maupun jalur alternatif, melalui

pengujian terhadap kombinasi cabang yang berbeda [3].

d) **Loop Testing**: Teknik loop testing memfokuskan pengujian pada struktur perulangan (loops) dalam kode. Loop testing mengidentifikasi kesalahan yang mungkin terjadi karena kesalahan pada perulangan seperti infinite loops atau kesalahan pada kondisi berhenti [1].

e) **Condition Coverage**: Teknik ini memfokuskan pengujian pada struktur perulangan (loops) dalam kode. Loop testing mengidentifikasi kesalahan yang mungkin terjadi karena kesalahan pada perulangan seperti infinite loops atau kesalahan pada kondisi berhenti [2].

3. DAFTAR REFERENSI

1. Myers, G. J., Sandler, C., & Badgett, T. (2011). *The Art of Software Testing*. Wiley.

2. ISTQB Foundation Level Syllabus, 2018

3. Kaner, Falk, & Nguyen, 1999, *Testing Computer Software*

4. Daftar Bacaan

1. Sama seperti pada daftar referensi

5. JADWAL PERKULIAHAN DAN TOPIK BAHASAN

Pertemuan Ke-	TOPIK BAHASAN	BACAAN
1	a. Kontrak Perkuliahan, Perkenalan dan Penjelasan b. Pengenalan Rekayasa Perangkat Lunak	Kontrak Perkuliahan
2	a. Karakteristik perangkat lunak b. Komponen perangkat lunak c. Model perangkat lunak d. Fungsi dan peran dari software engineer	1-6
3	a. Definisi SDLC b. Jenis-jenis SDLC	Idem
4	a. Observasi dan estimasi dalam perencanaan proyek b. Tujuan perencanaan proyek c. Manajemen proyek perangkat lunak yang efektif	Idem
5	a. Proses analisis kebutuhan b. Metode analisis kebutuhan c. Spesifikasi dan validasi kebutuhan	Idem
6	a. Perangkat bantu proses analisis kebutuhan b. Konsep dasar, Konteks, Proses, dan Prinsip Perancangan Perangkat Lunak;	Idem

	c. Isu mendasar dalam perancangan perangkat lunak	
7	a. Alat bantu perancangan (DFD dan UML) b. Macam-macam diagram yang terdapat pada UML (Class Diagram, Use Case Diagram, Activity Diagram, Sequence Diagram)	Idem
8	**UTS**	
9	Konsep dalam User Interface	Idem
10	a. Perencanaan dalam pengujian b. Proses testing: (black box testing, white box testing) c. Integration testing dan user testing d. Faults, Error dan Failures	Idem
11	Review Teknik Pengujian Perangkat Lunak dari proses testing	Idem
12	a. Pengujian unit b. Pengujian integrasi c. Pengujian sistem d. Debugging dan quality assurance	Idem
13	a. Quality assurance pada perangkat lunak b. Keamanan data akses	Idem
14	a. Definisi pemeliharaan perangkat lunak. b. Konsep Pemeliharaan Perangkat lunak	Idem
15	Teknik pemeliharaan perangkat lunak (Pemeliharaan korektif, pemeliharaan adaptif, pemeliharaan perfektif, pemeliharaan preventif)	Idem
16	**UAS**	