MODUL PEMBELAJARAN

Kurikulum Berbasis Luaran (Outcome Based Education)

Program Studi	S1 ILMU KOMPUTER	**Pertemuan Ke**	**12**
Mata Kuliah	Praktikum Pemrograman Web (4)		
Materi Pokok	**Pengujian dan Quality Assurance pada Aplikasi WebMetode pengujian dan jaminan kualitas perangkat lunak berbasis web.**		
Metode & Eval	Ceramah & SGD / Kognitif	**Bloom**	Evaluating
Sub-CPMK	Sub-CPMK043.2 - Mampu mengevaluasi efisiensi penggunaan teknologi web terbaru dalam konteks kebutuhan organisasi.		

Program Studi: S1 Ilmu Komputer

Praktikum Pemrograman Web

Pertemuan 12 Target: Evaluating

PENDAHULUAN

Dalam lanskap web global yang bergerak sangat cepat, kualitas aplikasi tidak lagi dipahami sekadar sebagai “aplikasi bisa dibuka” atau “fitur berjalan.” Kualitas kini berarti ketahanan sistem saat trafik melonjak, keamanan data pengguna lintas yurisdiksi, aksesibilitas bagi beragam pengguna, konsistensi pengalaman pada berbagai perangkat, serta efisiensi biaya pengembangan dan pemeliharaan. Bagi mahasiswa S1 Ilmu Komputer, pengujian dan quality assurance pada aplikasi web adalah fondasi intelektual sekaligus praktis untuk memastikan bahwa teknologi yang dibangun benar-benar bernilai bagi organisasi, bukan sekadar canggih secara teknis.

Secara filosofis, QA dalam rekayasa perangkat lunak web mencerminkan tanggung jawab ilmiah: setiap solusi harus dapat diuji, dibuktikan, direplikasi, dan dievaluasi dampaknya. Dalam konteks internasional, organisasi bersaing bukan hanya pada fitur, tetapi pada reliabilitas layanan digital, kepercayaan pengguna, dan kemampuan

merespons perubahan pasar. Karena itu, mahasiswa perlu memiliki cara berpikir evaluatif untuk menilai apakah teknologi web terbaru—misalnya framework front-end modern, arsitektur API, automated testing, atau cloud-based QA tools—benar-benar efisien terhadap kebutuhan organisasi yang spesifik.

Setelah mengikuti pertemuan ini, mahasiswa diharapkan mampu Sub-CPMK043.2 - Mampu mengevaluasi efisiensi penggunaan teknologi web terbaru dalam konteks kebutuhan organisasi.

Materi ini selaras dengan visi Program Studi S1 Ilmu Komputer yang menekankan penguasaan konsep, kemampuan analitis, dan penerapan teknologi secara bertanggung jawab. Di tingkat global, lulusan ilmu komputer dituntut mampu membaca problem organisasi secara sistemik: apakah sebuah teknologi mempercepat delivery, menurunkan defect rate, meningkatkan keamanan, atau justru menambah kompleksitas operasional. Oleh sebab itu, pengujian dan QA bukan sekadar tahap akhir, melainkan mekanisme penjamin mutu yang menyatu dengan siklus hidup pengembangan perangkat lunak modern.

PEMBAHASAN INTI

1. Hakikat Pengujian dan Quality Assurance dalam Aplikasi Web

- **Landasan teori dan filosofi:** Pengujian perangkat lunak adalah proses verifikasi dan validasi untuk memastikan sistem sesuai spesifikasi dan kebutuhan nyata pengguna. Quality Assurance (QA) lebih luas daripada testing karena mencakup pencegahan cacat melalui perencanaan proses, standar, audit, dan perbaikan berkelanjutan. Dalam perspektif rekayasa modern, kualitas bukan hasil kebetulan, melainkan hasil desain proses yang disiplin.
- **Konsep penting:**

> **Verification** berarti “apakah kita membangun produk dengan benar?” Sedangkan **validation** berarti “apakah kita membangun produk yang benar?” Analogi sederhananya: verification seperti memeriksa apakah resep diikuti dengan tepat, sedangkan validation seperti menilai apakah masakan itu benar-benar disukai pelanggan.

> **Defect** adalah kesalahan pada perangkat lunak. Bayangkan seperti retakan kecil pada jembatan: mungkin tidak terlihat saat awal, tetapi bisa

> membesar ketika beban pengguna meningkat.

- **Mekanisme teknis/prosedur kerja:**
  - Menyusun test plan berdasarkan requirement fungsional dan non-fungsional.
  - Mengidentifikasi test case untuk login, registrasi, transaksi, validasi input, dan responsivitas UI.
  - Menerapkan unit test, integration test, system test, regression test, dan user acceptance test.
  - Mendefinisikan acceptance criteria yang terukur, misalnya waktu respon < 2 detik pada 95% request.
- **Implementasi nyata, tantangan manajerial, dan hambatan lapangan:**
  - Organisasi sering terburu-buru merilis fitur sehingga QA dipersempit menjadi formalitas.
  - Tim produk, developer, dan QA kadang memiliki definisi “selesai” yang berbeda.
  - Hambatan umum: keterbatasan waktu, data uji yang tidak representatif, lingkungan staging yang tidak stabil, dan kurangnya dokumentasi requirement.
  - Manajerial: perlu kebijakan quality gate agar rilis tidak lolos sebelum memenuhi standar minimum.
- **Contoh kasus:** Pada portal akademik internasional, fitur KRS online harus tetap stabil saat periode registrasi dibuka serentak. QA memastikan sistem tidak gagal saat ribuan mahasiswa mengakses halaman yang sama.

2. Jenis-Jenis Pengujian Web dan Fungsi Strategisnya

- **Landasan teori dan filosofi:** Setiap jenis pengujian memiliki tujuan epistemologis yang berbeda. Unit test memeriksa bagian kecil, integration test memeriksa hubungan antarbagian, sedangkan end-to-end test menilai alur pengguna secara utuh. Dalam organisasi, pemilihan jenis test mencerminkan prioritas risiko.
- **Istilah teknis:**

> **Unit testing** adalah pengujian komponen terkecil, seperti memeriksa satu fungsi. Analogi: mengecek satu lampu sebelum memasangnya di seluruh rangkaian.

> **Regression testing** adalah pengujian ulang setelah perubahan kode untuk memastikan fitur lama tidak rusak. Analogi: setelah memperbaiki atap rumah, Anda mengecek apakah dinding dan plafon tetap aman.

> **End-to-end testing** menguji perjalanan lengkap pengguna dari awal hingga akhir. Analogi: mencoba seluruh proses belanja dari memilih barang sampai pembayaran berhasil.

- **Mekanisme teknis/prosedur kerja:**
  - Unit test dijalankan otomatis menggunakan framework seperti Jest, PHPUnit, atau PyTest.
  - Integration test memeriksa koneksi antar modul, misalnya front-end ke API dan API ke database.
  - UI test memastikan komponen visual tampil dan berfungsi pada browser berbeda.
  - Performance test mengukur throughput, latency, dan resource usage.
  - Security test mengidentifikasi celah seperti XSS, CSRF, dan SQL injection.
- **Implementasi nyata, tantangan manajerial, dan hambatan lapangan:**
  - Automasi test membutuhkan investasi awal pada tooling, script, dan maintenance.
  - Test yang terlalu banyak tetapi tidak relevan dapat menambah biaya tanpa meningkatkan kualitas secara signifikan.
  - Manajer proyek harus menyeimbangkan cakupan test dengan batas waktu dan anggaran.
  - Ketergantungan pada browser, plugin, dan perangkat berbeda sering menimbulkan hasil yang tidak konsisten.
- **Contoh kasus:** Aplikasi e-commerce global harus diuji pada Chrome, Safari, dan Firefox karena perilaku CSS dan JavaScript dapat berbeda. Kegagalan satu browser dapat menurunkan konversi penjualan secara signifikan.

3. Quality Assurance sebagai Sistem, Bukan Sekadar Aktivitas Akhir

- **Landasan teori dan filosofi:** QA modern berakar pada prinsip continuous improvement. Kualitas dibangun sejak requirement, desain, coding, hingga deployment. Ini sejalan dengan pemikiran Deming tentang perbaikan proses berkelanjutan dan pengurangan variasi.

- **Istilah teknis:**

> **Continuous Integration (CI)** adalah praktik menggabungkan kode secara rutin ke repositori utama dan menjalankan test otomatis. Analogi: setiap orang di dapur menambahkan bahan sedikit demi sedikit lalu langsung mencicipi apakah rasanya masih seimbang.

> **Continuous Delivery/Deployment (CD)** berarti perubahan yang lolos test dapat dirilis dengan cepat dan aman. Analogi: jalur produksi yang memastikan barang siap kirim kapan pun kualitasnya lolos inspeksi.

- **Mekanisme teknis/prosedur kerja:**
  - Integrasi QA ke pipeline CI/CD.
  - Penggunaan linting, static code analysis, dan code review sebagai pencegahan awal cacat.
  - Monitoring pascadeploy untuk mendeteksi error, crash, dan penurunan performa.
  - Penerapan defect tracking agar masalah terdokumentasi dan tidak berulang.
- **Implementasi nyata, tantangan manajerial, dan hambatan lapangan:**
  - Budaya organisasi sering memisahkan QA dari development, padahal kualitas harus menjadi tanggung jawab bersama.
  - Kurangnya disiplin dokumentasi menyebabkan defect yang sama muncul berulang.
  - Tekanan deadline dapat mendorong shortcut teknis yang berbahaya bagi stabilitas jangka panjang.
  - Perlu kepemimpinan teknis yang mampu memprioritaskan kualitas sebagai investasi, bukan biaya tambahan.
- **Contoh kasus:** Startup fintech yang menambahkan fitur pembayaran baru wajib memastikan integrasi dengan payment gateway, audit log, dan notifikasi berjalan konsisten. Satu bug kecil dapat berdampak pada transaksi ganda atau kegagalan pencatatan keuangan.

4. Evaluasi Efisiensi Teknologi Web Terbaru untuk Kebutuhan Organisasi

- **Landasan teori dan filosofi:** Efisiensi tidak hanya berarti cepat, tetapi juga tepat guna, hemat sumber daya, mudah dipelihara, aman, dan sesuai konteks

organisasi. Teknologi terbaru harus dinilai berdasarkan value proposition, bukan popularitas.

- **Istilah teknis:**

> **Framework** adalah kerangka kerja yang menyediakan struktur dan komponen siap pakai. Analogi: rangka rumah yang mempercepat pembangunan, tetapi tetap harus disesuaikan dengan ukuran lahan dan kebutuhan penghuni.

> **Scalability** adalah kemampuan sistem tumbuh mengikuti beban pengguna. Analogi: jalan tol yang bisa menampung lebih banyak kendaraan tanpa macet parah.

> **Technical debt** adalah "utang teknis" akibat keputusan cepat yang membuat perawatan masa depan lebih sulit. Analogi: menambal kebocoran dengan lakban sementara; murah sekarang, mahal nanti.

- **Mekanisme teknis/prosedur kerja:**
  - Membandingkan teknologi berdasarkan kecepatan development, learning curve, komunitas, dokumentasi, dan biaya operasional.
  - Melakukan proof of concept sebelum adopsi penuh.
  - Mengukur metrik seperti response time, defect density, maintainability index, dan deployment frequency.
  - Menilai kompatibilitas teknologi dengan infrastruktur organisasi dan kompetensi tim.
- **Implementasi nyata, tantangan manajerial, dan hambatan lapangan:**
  - Teknologi baru sering diadopsi karena tren, bukan karena kebutuhan nyata.
  - Organisasi yang terlalu cepat berpindah stack dapat kehilangan stabilitas dan produktivitas tim.
  - Manajer perlu mempertimbangkan total cost of ownership, bukan hanya biaya lisensi atau hosting awal.
  - Hambatan umum: migrasi data, kompatibilitas modul lama, dan resistensi pengguna internal.

- **Contoh kasus:** Universitas yang ingin mengganti sistem akademik lama ke framework web modern harus menilai apakah peningkatan performa sebanding dengan biaya migrasi, pelatihan staf, dan risiko downtime pada masa transisi.

5. Standar Kualitas Internasional dan Perspektif Global

- **Landasan teori dan filosofi:** Dalam ekosistem global, kualitas perangkat lunak tidak lepas dari standar internasional. Standar membantu organisasi menyamakan bahasa mutu, meminimalkan subjektivitas, dan memperkuat akuntabilitas.
- **Istilah teknis:**

> **ISO/IEC 25010** adalah model kualitas perangkat lunak yang mencakup functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, dan portability. Analogi: daftar pemeriksaan kualitas mobil, bukan hanya apakah mesinnya hidup.

> **Accessibility** berarti aplikasi dapat digunakan oleh semua orang, termasuk pengguna dengan keterbatasan fisik atau sensorik. Analogi: pintu masuk gedung yang ramah kursi roda dan jelas petunjuknya.

- **Mekanisme teknis/prosedur kerja:**
  - Menggunakan checklist kualitas berbasis ISO/IEC 25010.
  - Melakukan audit aksesibilitas dengan WCAG sebagai acuan.
  - Meninjau keamanan dan privasi sesuai regulasi lintas negara, seperti GDPR untuk konteks Eropa.
  - Menyesuaikan bahasa, format waktu, dan mata uang untuk pengguna global.
- **Implementasi nyata, tantangan manajerial, dan hambatan lapangan:**
  - Organisasi multinasional harus menyeimbangkan standardisasi global dan kebutuhan lokal.
  - Compliance dapat meningkatkan biaya awal, namun mengurangi risiko hukum dan reputasi.
  - Hambatan utama adalah kesenjangan pengetahuan tim terhadap standar internasional.
  - Perlu pelatihan berkelanjutan agar QA tidak hanya bersifat administratif.
- **Contoh kasus:** Platform pembelajaran global harus memastikan subtitle, navigasi keyboard, dan kontras warna memadai agar dapat diakses mahasiswa

dari berbagai negara dan latar belakang kemampuan.

6. Integrasi Pengujian, QA, dan Pengambilan Keputusan Organisasi

- **Landasan teori dan filosofi:** Evaluasi teknologi web terbaru harus berbasis bukti. Dalam ilmu komputer, keputusan yang baik lahir dari data pengujian, bukan asumsi atau preferensi pribadi. Ini adalah bentuk literasi teknis yang matang.
- **Istilah teknis:**

> **Evidence-based decision making** adalah pengambilan keputusan berbasis bukti. Analogi: dokter memilih terapi setelah melihat hasil pemeriksaan, bukan karena tebakan.

> **Benchmarking** adalah membandingkan performa sistem dengan standar atau kompetitor. Analogi: pelari mengukur waktu tempuhnya dibanding target atau lawan tanding.

- **Mekanisme teknis/prosedur kerja:**
  - Mengumpulkan hasil test, log, metrik performa, dan feedback pengguna.
  - Menganalisis trade-off antara kecepatan pengembangan, kualitas, dan biaya.
  - Menentukan apakah teknologi baru layak diadopsi, dimodifikasi, atau ditolak.
  - Mendokumentasikan alasan keputusan sebagai dasar audit dan pembelajaran organisasi.
- **Implementasi nyata, tantangan manajerial, dan hambatan lapangan:**
  - Keputusan teknologi sering dipengaruhi faktor non-teknis seperti vendor, tren pasar, atau tekanan pimpinan.
  - Mahasiswa perlu belajar mengkritisi klaim “lebih cepat” atau “lebih modern” dengan data nyata.
  - Organisasi yang matang memiliki komite arsitektur atau review board untuk menilai perubahan besar.
  - Hambatan umum adalah bias konfirmasi dan kurangnya data historis.
- **Contoh kasus:** Sebuah perusahaan logistik global mempertimbangkan migrasi dari monolit ke microservices. Keputusan harus didasarkan pada hasil load testing, kompleksitas operasional, kemampuan tim DevOps, dan kebutuhan skalabilitas jangka panjang.

ANALISIS KOMPARATIF GLOBAL

Kategori Analisis	Perspektif Ideal (Global)	Realita & Gap Lapangan
Automasi Testing	Mayoritas pengujian kritis terotomasi dalam pipeline CI/CD untuk mempercepat rilis dan menekan defect.	Banyak organisasi masih mengandalkan testing manual sehingga lambat dan rawan human error.
Standar Kualitas	Quality gate mengacu pada ISO/IEC 25010, OWASP, dan WCAG untuk kualitas, keamanan, dan aksesibilitas.	Standar sering dipahami sebagai dokumen administratif, bukan alat evaluasi teknis yang hidup.
Evaluasi Teknologi Baru	Adopsi teknologi didasarkan pada proof of concept, benchmark, dan analisis biaya-manfaat.	Keputusan sering dipengaruhi tren, vendor, atau preferensi individu tanpa bukti memadai.
Budaya QA	QA menjadi tanggung jawab lintas fungsi: developer, tester, product owner, dan manajer.	QA masih sering diposisikan sebagai “polisi akhir” yang bekerja setelah coding selesai.
Aksesibilitas & Global Reach	Aplikasi dirancang inklusif, multibahasa, dan patuh regulasi lintas negara.	Fitur aksesibilitas dan lokalisasi sering diabaikan karena dianggap tidak mendesak.

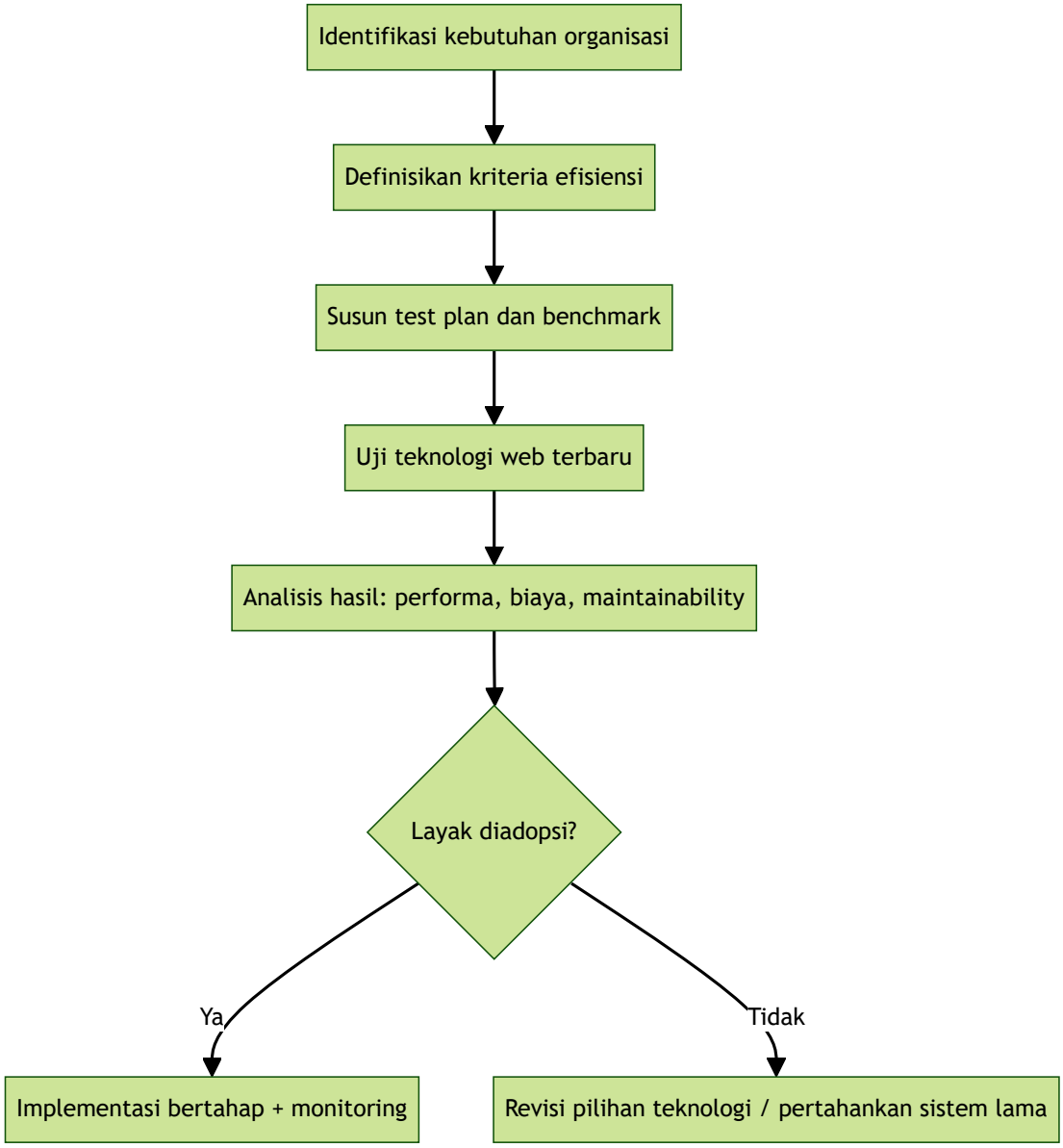
STUDI KASUS SPESIFIK: S1 Ilmu Komputer

Kasus: Sebuah universitas internasional ingin mengadopsi framework front-end terbaru untuk portal akademik mahasiswa asing. Tim pengembang mengklaim framework tersebut lebih cepat, lebih modern, dan lebih mudah dipelihara. Namun pimpinan fakultas meminta bukti bahwa adopsi teknologi ini benar-benar efisien bagi kebutuhan organisasi.

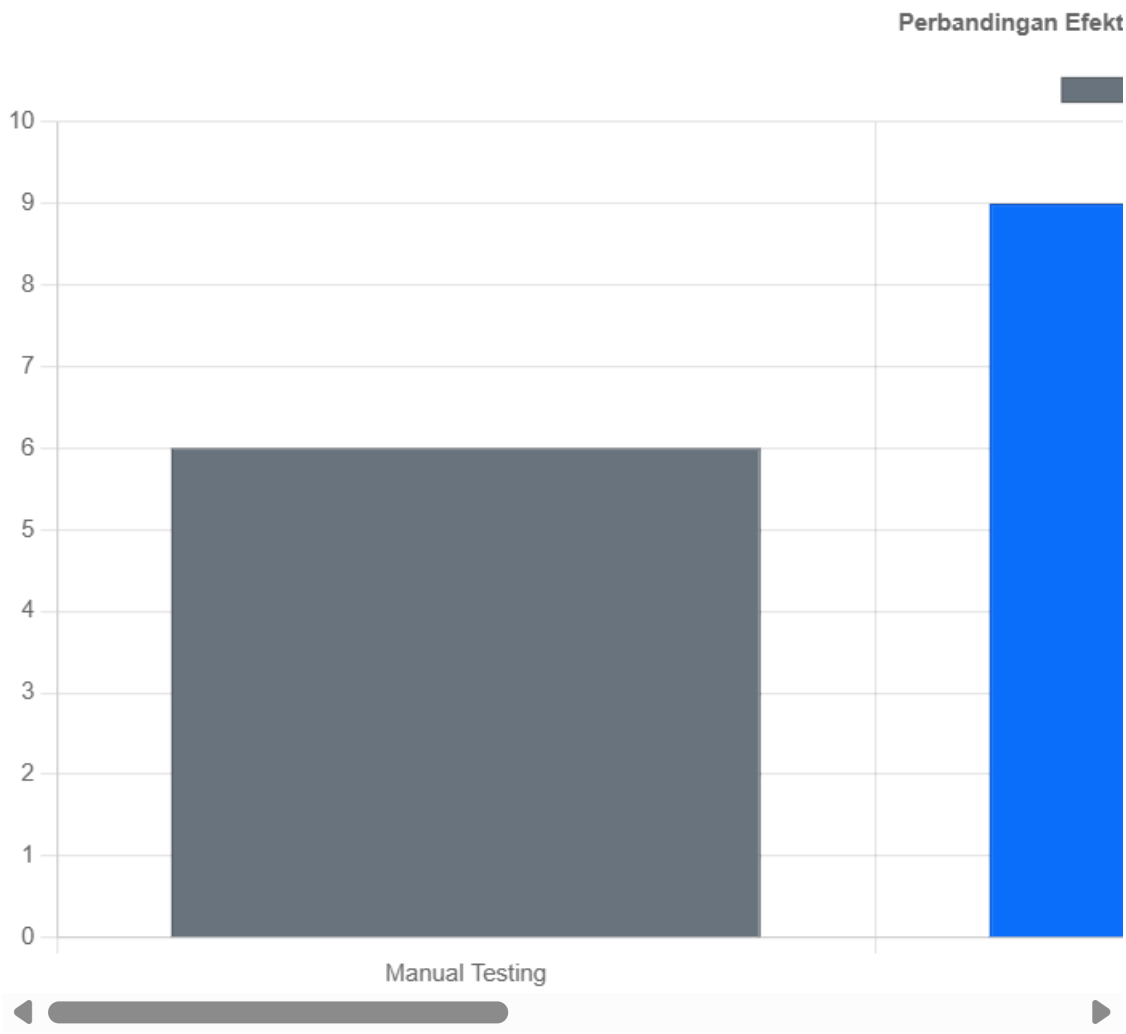
Analisis kritis yang harus dilakukan mahasiswa:

- Membandingkan waktu development sebelum dan sesudah penggunaan framework baru.
- Mengukur performa halaman pada perangkat mobile dan desktop.
- Menilai kemudahan maintenance berdasarkan struktur komponen dan dokumentasi.
- Menguji kompatibilitas dengan API akademik, sistem autentikasi, dan database lama.
- Menilai dampak terhadap aksesibilitas mahasiswa internasional, termasuk dukungan bahasa dan navigasi keyboard.

Visualisasi proses evaluasi:



Grafik data perbandingan efektivitas metode pengujian:



Interpretasi: Dalam banyak organisasi, automated testing biasanya unggul dalam kecepatan dan konsistensi, tetapi manual testing tetap penting untuk mengevaluasi pengalaman pengguna, alur bisnis yang kompleks, dan aspek yang sulit diotomasi. Mahasiswa S1 Ilmu Komputer harus mampu menilai kombinasi metode yang paling efisien, bukan memilih satu metode secara absolut.

> **Proof of concept (PoC)** adalah percobaan kecil untuk membuktikan ide atau teknologi sebelum dipakai luas. Analogi: mencoba satu alat baru di dapur sebelum mengganti seluruh peralatan restoran.

PANDUAN AKTIVITAS & ASESMEN

Metode pembelajaran: Ceramah & SGD

- **Ceramah terarah:** Dosen menjelaskan konsep testing, QA, standar kualitas, dan evaluasi efisiensi teknologi web terbaru dengan contoh global.
- **SGD (Small Group Discussion):** Mahasiswa dibagi dalam kelompok kecil untuk membandingkan dua pendekatan pengujian atau dua teknologi web, lalu menyimpulkan mana yang paling efisien untuk kebutuhan organisasi tertentu.
- **Aktivitas kelas:**
  - Menganalisis satu studi kasus aplikasi web yang gagal karena lemahnya QA.
  - Menyusun matriks keputusan sederhana: manfaat, risiko, biaya, waktu, dan maintainability.
  - Menyampaikan argumen berbasis data, bukan opini semata.

Instrumen Asesmen HOTS Level Evaluating

1. **Tugas analisis komparatif:** Mahasiswa membandingkan dua metode testing pada aplikasi web dan menilai metode mana yang lebih efisien untuk skenario organisasi tertentu.
2. **Rubrik evaluasi:**
   - Ketepatan identifikasi kriteria efisiensi.
   - Kekuatan argumentasi berbasis data pengujian.
   - Kemampuan menimbang trade-off antara performa, biaya, dan maintainability.
   - Kejelasan kesimpulan dan justifikasi keputusan.
3. **Bentuk asesmen:** Esai analitis, presentasi kelompok, dan diskusi kritis berbasis studi kasus.
4. **Indikator keberhasilan:** Mahasiswa mampu memberikan rekomendasi teknologi atau metode pengujian yang rasional, terukur, dan sesuai konteks organisasi global.

Referensi (APA Style):

1. Arcuri, A., & Briand, L. (2011). A practical guide for using statistical tests to assess randomized algorithms in software engineering. *Proceedings of the 33rd International Conference on Software Engineering*, 1–10.
2. Jorgensen, P. C. (2013). *Software testing: A craftsman's approach* (4th ed.). CRC Press.

3. ISO/IEC. (2011). *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models (ISO/IEC 25010:2011)*. International Organization for Standardization.
4. Myers, G. J., Sandler, C., & Badgett, T. (2011). *The art of software testing* (3rd ed.). Wiley.
5. OWASP Foundation. (2021). *OWASP Web Security Testing Guide*. https://owasp.org