**1. Functional Suitability (Kesesuaian Fungsional)**

Ini adalah karakteristik yang paling dasar.

* **Definisi Sederhana:** "Apakah software ini melakukan apa yang seharusnya ia lakukan?"
* **Fokus:** Memastikan semua fitur berjalan sesuai spesifikasi dan kebutuhan pengguna.

**🤝 Mari Kerjakan Bersama (Studi Kasus SI-PERPUS):**

**A. Functional Completeness (Kelengkapan Fungsional)**

* **Pertanyaan Penguji:** Apakah *semua* fitur yang dijanjikan ada?
* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Baik, mari kita mulai dari yang paling dasar. Melihat daftar fitur SI-PERPUS tadi, coba sebutkan tombol, menu, atau halaman apa saja yang *wajib* ada di sistem kita agar bisa disebut 'lengkap'?"

* **✅ Contoh Jawaban Mahasiswa:**
  + "Harus ada field untuk input username dan password, Pak. Sama tombol 'Login'."
  + "Setelah login, harus ada halaman utama atau dashboard."
  + "Di halaman utama, wajib ada kotak pencarian buku dan tombol 'Cari'."
  + "Untuk Pustakawan, harus ada menu untuk validasi peminjaman dan pengembalian buku."
  + "Mahasiswa harus punya halaman untuk melihat riwayat peminjaman dan jumlah denda."

**B. Functional Correctness (Kebenaran Fungsional)**

* **Pertanyaan Penguji:** Apakah fitur yang ada memberikan hasil yang benar?
* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Sekarang, mari kita bicara kebenaran. Misalkan, pada fitur perhitungan denda, skenarionya adalah denda Rp1.000/hari. Jika seorang mahasiswa terlambat 5 hari, apa output yang benar? Nah, sekarang coba kalian pikirkan, skenario salah seperti apa yang bisa terjadi di fitur ini? Apa saja yang bisa membuat perhitungannya tidak benar?"

* **✅ Contoh Jawaban Mahasiswa:**
  + "Bisa jadi sistemnya salah hitung karena hari libur seperti Minggu atau tanggal merah ikut dihitung, padahal perpustakaan tutup."
  + "Kalau mahasiswa pinjam 3 buku dan semuanya telat, bisa jadi sistem hanya menghitung denda untuk 1 buku saja."
  + "Mungkin ada bug pembulatan angka, jadi dendanya Rp 2.999, bukan Rp 3.000."
  + "Kesalahan bisa terjadi karena zona waktu server berbeda, Pak. Jadi pengembalian jam 1 pagi sudah dianggap telat sehari."

**C. Functional Appropriateness (Kepatutan Fungsional)**

* **Pertanyaan Penguji:** Apakah fitur-fitur tersebut membantu pengguna mencapai tujuannya dengan tepat?
* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Di SI-PERPUS ada dua jenis pengguna: Mahasiswa dan Pustakawan. Menurut kalian, fitur 'Validasi Peminjaman' itu pantasnya bisa diakses oleh siapa? Kenapa? Selain itu, fitur apa lagi yang hanya boleh diakses Pustakawan? Dan fitur apa yang hanya relevan untuk Mahasiswa?"

* **✅ Contoh Jawaban Mahasiswa:**
  + "Validasi peminjaman jelas hanya untuk Pustakawan, Pak. Kalau mahasiswa bisa validasi sendiri, nanti bisa pinjam buku tanpa izin."
  + "Fitur menambah, mengubah, atau menghapus data buku juga harusnya hanya untuk pustakawan."
  + "Fitur untuk melihat laporan peminjaman bulanan juga untuk pustakawan, untuk keperluan laporan."
  + "Untuk mahasiswa, yang relevan itu fitur cek status buku yang sedang dipinjam dan melihat histori peminjaman pribadi."

**2. Performance Efficiency (Efisiensi Kinerja)**

* **Definisi Sederhana:** "Seberapa cepat dan hemat software ini saat dijalankan?"
* **Fokus:** Kecepatan respon, penggunaan sumber daya (CPU, RAM), dan kapasitas.

**🤝 Mari Kerjakan Bersama (Studi Kasus SI-PERPUS):**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Oke, sekarang bayangkan kalian adalah pengguna SI-PERPUS. Saat kalian mencari sebuah buku, berapa lama waktu tunggu (loading) yang menurut kalian masih wajar dan bisa diterima? 1 detik? 3 detik? 10 detik?" ... "(Dosen) Lanjutan. Bayangkan sistem kita tiba-tiba jadi sangat lambat saat jam istirahat siang. Menurut kalian, sebagai seorang analis mutu, apa kira-kira penyebabnya?"

* **✅ Contoh Jawaban Mahasiswa:**
  + **(Waktu Tunggu):** "Menurut saya di bawah 3 detik, Pak. Kalau lebih dari itu biasanya pengguna tidak sabar dan menutup aplikasi."
  + **(Penyebab Lambat):** "Karena pada jam istirahat, banyak sekali mahasiswa yang mengakses sistem secara bersamaan untuk mencari buku."
  + **(Penyebab Lambat):** "Bisa jadi karena *query* ke databasenya tidak efisien. Jadi saat banyak yang mencari, databasenya jadi 'stres' dan lambat merespon."
  + **(Penyebab Lambat):** "Mungkin juga spesifikasi servernya terlalu kecil, Pak. RAM atau CPU-nya tidak cukup untuk melayani banyak pengguna sekaligus."

**3. Usability (Kebergunaan)**

* **Definisi Sederhana:** "Seberapa mudah dan nyaman software ini digunakan oleh pengguna?"
* **Fokus:** Tampilan antarmuka yang intuitif, kemudahan dipelajari, dan perlindungan dari kesalahan pengguna.

**🤝 Mari Kerjakan Bersama (Studi Kasus SI-PERPUS):**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Sekarang, mari kita semua bayangkan kita baru pertama kali membuka aplikasi SI-PERPUS. Tidak ada buku panduan. Saya ingin mencari buku 'Laskar Pelangi'. Coba deskripsikan langkah-langkah yang akan kalian lakukan." ... "(Dosen) Skenario lain: Seorang pustakawan salah memasukkan NIM mahasiswa. Menurut kalian, apa respons sistem yang baik? Coba buatkan satu contoh pesan error yang baik!"

* **✅ Contoh Jawaban Mahasiswa:**
  + **(Langkah Pencarian):** "Pertama, saya pasti akan mencari kotak input teks yang ada tulisan 'Cari Buku' atau ada ikon kaca pembesar. Lalu saya ketik judulnya di sana dan tekan Enter atau klik tombol Cari."
  + **(Pesan Error Baik):** "Contoh yang baik: **'NIM tidak ditemukan. Mohon periksa kembali 10 digit NIM yang Anda masukkan.'**"
  + **(Pesan Error Baik):** "Bisa juga ditambahkan petunjuk, misalnya **'Error: Format tanggal salah. Silakan gunakan format HH/BB/TTTT (contoh: 19/09/2025).'**"
  + **(Pesan Error Buruk):** "Kalau yang jelek itu yang cuma menampilkan tulisan 'Error: Input tidak valid' atau 'Error 400', Pak. Pengguna jadi bingung salahnya di mana."

*(Pola yang sama diterapkan untuk karakteristik lainnya)*

**4. Reliability (Keandalan)**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Mari kita bermain skenario 'bencana'. Bayangkan, saat pustakawan sedang sibuk-sibuknya memproses 10 peminjaman buku, tiba-tiba server perpustakaan mati listrik selama 5 detik lalu hidup lagi. Apa hal terburuk yang bisa terjadi pada data peminjaman tersebut? Dan apa yang seharusnya dilakukan oleh sistem yang 'andal'?"

* **✅ Contoh Jawaban Mahasiswa:**
  + **"Hal terburuknya, data peminjaman yang sedang diproses tadi hilang sama sekali, jadi tidak tercatat di sistem."**
  + **"Bisa lebih parah, Pak. Databasenya bisa *corrupt* atau rusak karena proses penulisan datanya terputus di tengah jalan."**
  + **"Sistem yang andal seharusnya punya mekanisme *transaksi*. Jadi kalau prosesnya gagal di tengah jalan, semua perubahan akan dibatalkan (di-*rollback*), sehingga data tetap aman."**
  + **"Atau, sistem bisa menyimpan data sementara secara lokal, dan otomatis mencoba mengirimnya lagi ke server saat server sudah kembali normal."**

**5. Security (Keamanan)**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Sekarang, ayo kita coba berpikir seperti 'hacker baik'. Saya adalah mahasiswa A. Saya ingin tahu buku apa saja yang pernah dipinjam oleh mahasiswa B. Menurut kalian, celah apa yang bisa saya coba manfaatkan di sistem SI-PERPUS untuk mendapatkan info itu?"

* **✅ Contoh Jawaban Mahasiswa:**
  + **"Saya akan coba lihat URL di browser, Pak. Kalau URL profil saya adalah .../profil?id=123, saya akan coba ganti jadi .../profil?id=124 untuk melihat profil mahasiswa lain."**
  + **"Mencoba menebak password akun Pustakawan. Biasanya passwordnya lemah seperti 'admin' atau 'pustaka123'."**
  + **"Kalau ada fitur lupa password, saya coba manfaatkan untuk mereset password orang lain kalau pertanyaannya mudah ditebak."**

**6. Maintainability (Kemudahan Pemeliharaan)**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Bayangkan kalian adalah programmer baru di tim SI-PERPUS. Kalian diberi tugas: 'Tolong tambahkan fitur denda untuk buku yang rusak, bukan hanya terlambat'. Jika kode sistem ini ditulis dengan baik (modular), kira-kira di bagian file atau folder mana kalian akan mulai mencari kode yang relevan dengan 'denda'?"

* **✅ Contoh Jawaban Mahasiswa:**
  + **"Saya akan cari file atau folder yang namanya berhubungan, seperti 'peminjaman', 'transaksi', atau mungkin ada folder khusus 'kalkulasi'."**
  + **"Kalau kodenya bagus, harusnya ada file spesifik bernama DendaService.java atau penalty\_calculator.py. Jadi saya langsung tahu harus mengubah di mana."**
  + **"Pentingnya dipisah agar saat saya mengubah logika denda, saya tidak secara tidak sengaja merusak fitur lain seperti pencarian buku."**

**7. Portability (Portabilitas)**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Pihak kampus memutuskan untuk pindah dari server milik sendiri (on-premise) ke layanan Cloud. Sebagai tim teknis, tantangan apa saja yang mungkin kita hadapi saat 'memindahkan' aplikasi SI-PERPUS kita?"

* **✅ Contoh Jawaban Mahasiswa:**
  + **"Versi PHP atau Java di server cloud mungkin berbeda dengan di server lama, jadi ada kode yang harus disesuaikan."**
  + **"Konfigurasi database, Pak. Cara koneksi ke database di cloud pasti berbeda dengan koneksi di server lokal."**
  + **"Settingan *environment variable* atau path folder pasti berubah, jadi harus dikonfigurasi ulang."**

**8. Compatibility (Kompabilitas)**

* **💡 Titik Interaksi Mahasiswa:**

"(Dosen) Untuk membuat hidup pustakawan lebih mudah, kita ingin SI-PERPUS bisa terhubung dengan Sistem Informasi Akademik (SIAKAD). Apa keuntungan jika kedua sistem ini bisa 'berbicara' satu sama lain? Informasi apa yang bisa mereka pertukarkan?"

* **✅ Contoh Jawaban Mahasiswa:**
  + **"Keuntungannya, data mahasiswa di SI-PERPUS bisa otomatis ter-update dari SIAKAD. Jadi Pustakawan tidak perlu input data mahasiswa baru secara manual."**
  + **"SIAKAD bisa otomatis mengecek ke SI-PERPUS untuk syarat bebas pustaka saat mahasiswa mau wisuda. Jadi prosesnya lebih cepat."**
  + **"Bisa diterapkan *Single Sign-On* (SSO). Mahasiswa cukup login sekali di SIAKAD, dan otomatis bisa masuk ke SI-PERPUS tanpa harus login lagi."**