Tipe-Tipe Requirement
dari Business sampai Non-Functional

Muhammad Riduwan Khafidi
Program Studi Teknik Rekayasa Perangkat Lunak
Politeknik Negeri Batam
4342611034@students.polibatam.ac.id

November 5, 2026

Abstrak

Makalah ini membahas empat tipe requirement yang wajib dibedakan seorang analis: business, user, functional, dan non-functional. Ia tidak hanya menjelaskan definisi, tapi memberi alat uji praktis berupa diagram keputusan dan tabel perbandingan agar pembaca bisa mengklasifikasi requirement yang ditemukan di lapangan. Tujuh visualisasi dan tiga tabel menyertai pembahasan, termasuk enam dimensi pembentuk user class yang tidak selalu disebut di kelas. Sebuah studi kasus sepuluh langkah dengan sistem penjadwalan kuliah menunjukkan bagaimana keempat tipe ini saling berhubungan dan menghasilkan artefak konkret. Konsep traceability sebagai jembatan antara business need dan test case juga diperkenalkan sebagai pelengkap. Lembar rujukan cepat di akhir merangkum istilah dan template yang wajib dikuasai sebelum pertemuan kelima.

1 Pendahuluan

1.1 Analogi Pembuka

Bayangkan sebuah restoran. Di meja tamu ada menu dengan foto dan harga. Di dapur ada resep dengan takaran bahan dan urutan langkah. Di ruang manajer ada target pendapatan bulanan dan daftar supplier.

Ketiganya berbicara tentang makanan yang sama, tapi di level yang berbeda. Menu berbicara tentang **apa yang dijual**. Resep berbicara tentang **bagaimana memasaknya**. Target manajer berbicara tentang **mengapa restoran ini ada**. Kalau ketiga dokumen itu dicampur, dapur akan menerima menu sebagai instruksi masak, dan manajer akan menerima resep sebagai laporan keuangan.

Requirement juga hidup di beberapa level seperti itu. Business requirement adalah target manajer. User requirement adalah menu. System requirement adalah resep. Mencampur level-level ini adalah salah satu kesalahan paling sering dalam penulisan requirement.

1.2 Mengapa Topik Ini Penting

Pada minggu-minggu sebelumnya, kita mempelajari apa itu requirement, siapa stakeholder-nya, dan bagaimana analyst menjalankan perannya. Di minggu ini, kita belajar bahwa "requirement" bukan satu benda. Ia adalah keluarga empat tipe yang saling bergantung. Kalau salah satu tipe hilang, atau dua tipe dicampur, dokumen requirement menjadi tidak bisa diuji dan tidak bisa dilacak.

1.3 Peta Pembahasan

Makalah ini disusun sebagai berikut. Bagian kedua memetakan empat level requirement dalam satu hierarki. Bagian ketiga sampai keenam membahas satu per satu: business, user, functional, dan non-functional. Bagian ketujuh memberi alat uji klasifikasi. Bagian kedelapan sampai kesepuluh merangkum perbandingan, kesalahan umum, dan lembar rujukan cepat. Bagian kesebelas memandu studi kasus sepuluh langkah dengan sistem penjadwalan kuliah. Bagian kedua belas menutup dengan tiga prinsip.

2 Peta Empat Level Requirement

Requirement hidup di empat level. Perhatikan Gambar 1. Semakin ke atas, semakin abstrak dan semakin dekat ke tujuan bisnis. Semakin ke bawah, semakin konkret dan semakin dekat ke kode.

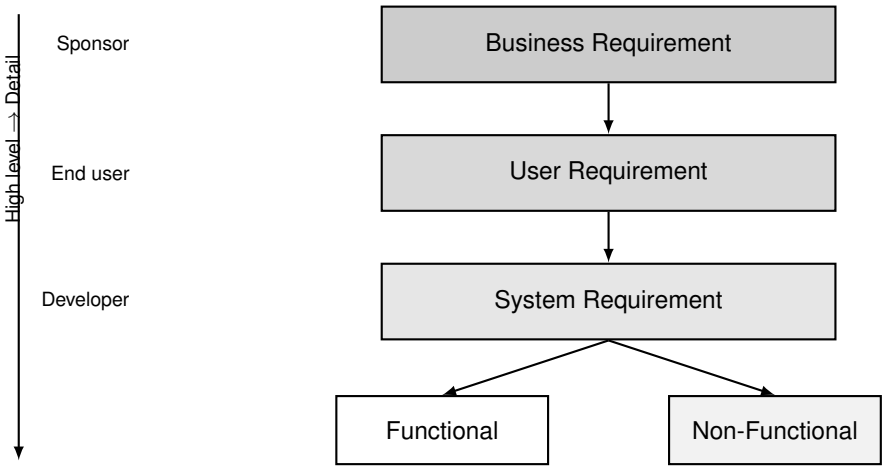


Figure 1. Empat level requirement dan siapa yang membacanya.

Tiga hal yang perlu diperhatikan dari hierarki ini.

Pertama, setiap level punya pembaca utama yang berbeda. Business requirement dibaca sponsor. User requirement dibaca end user. System requirement dibaca developer. Menulis business requirement dalam bahasa developer adalah kesalahan yang setara dengan menulis menu restoran dalam bahasa koki.

Kedua, setiap level harus bisa dilacak ke level di atasnya. Setiap functional requirement wajib

bisa dilacak balik ke satu user requirement, dan user requirement wajib bisa dilacak balik ke satu business requirement. Kalau tidak, requirement itu tidak punya alasan untuk ada.

Ketiga, system requirement adalah satu-satunya yang dipecah dua. Business dan user requirement cukup satu lapis. System requirement harus dipecah menjadi functional dan non-functional, karena keduanya diuji dengan cara yang berbeda.

Pelacakan antar level ini disebut **traceability**. Ia tidak selalu eksplisit di slide, tapi tanpa traceability, dokumen requirement hanya tumpukan kalimat.

3 Business Requirement

Business requirement menjelaskan **mengapa** organisasi melakukan proyek ini. Apa yang bisnis coba capai dengan proyek ini. Ia mencakup visi dan scope proyek, dan biasanya dinyatakan dalam project charter, business case, atau vision and scope statement.

Business requirement dikumpulkan dari project owner, stakeholder, dan tim proyek. Tujuannya satu: menyusun tujuan tingkat tinggi proyek. Tanpa informasi ini, tim tidak akan tahu mengapa mereka membangun sesuatu.

Tapi perlu dicatat: **kita tidak bisa membangun software hanya dari informasi tingkat tinggi ini**. Business requirement adalah arah, bukan rencana. Ia memberi tahu kita ke mana harus pergi, bukan jalan mana yang harus diambil.

Ada tujuh dokumen yang biasa dipakai untuk mengomunikasikan business requirement: problem statement, project vision, project constraints (budget, schedule, resources), project objectives, project scope statements, business process analysis, dan stakeholder analysis.

Yang paling sering dilupakan adalah **business process analysis**. Tujuannya menentukan bagaimana proses bisnis akan berjalan. Sebelum mengotomasi proses, kita wajib memahaminya dulu. Perhatikan Gambar 2.

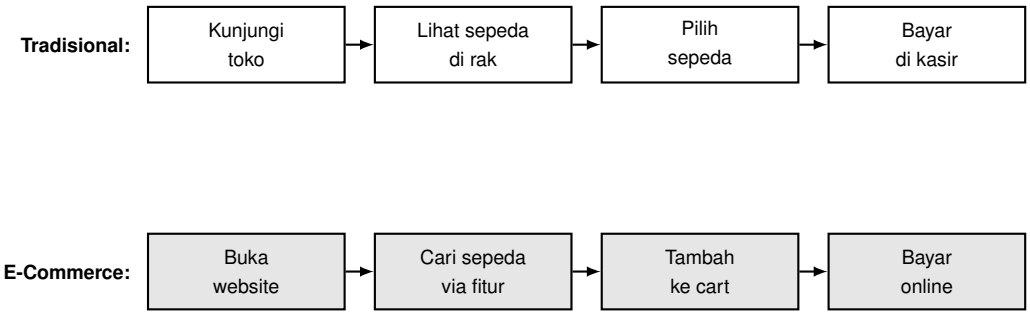


Figure 2. Business process tradisional dan versi otomasi untuk pembelian sepeda.

Perhatikan bahwa alur e-commerce bukan sekadar alur tradisional yang dipindahkan ke layar. Setiap langkah berubah maknanya. “Kunjungi toko” menjadi “buka website”. “Lihat sepeda di rak” menjadi “cari sepeda via fitur pencarian”. “Bayar di kasir” menjadi “bayar online”. Bisnis proses yang tidak dipahami sebelum diotomasi akan menghasilkan sistem yang memaksa user bekerja dengan cara mesin, bukan sebaliknya.

4 User Requirement

User requirement menguraikan apa saja yang bisa dilakukan user dengan produk, dan mendefinisikan apa yang produk harus lakukan untuk user. User adalah individu yang akan memakai software kita. Mereka sering disebut end user.

End user sangat kritis bagi keberhasilan produk. Itulah sebabnya user requirement adalah salah satu requirement paling penting. Bentuknya bisa use case, user story, atau scenario.

4.1 Enam Dimensi User Class

Tidak semua user sama. Menganggap semua user identik adalah kesalahan yang menghasilkan produk yang tidak nyaman untuk siapa pun. User bisa dikelompokkan menjadi beberapa user class berdasarkan enam dimensi. Perhatikan Gambar 3.

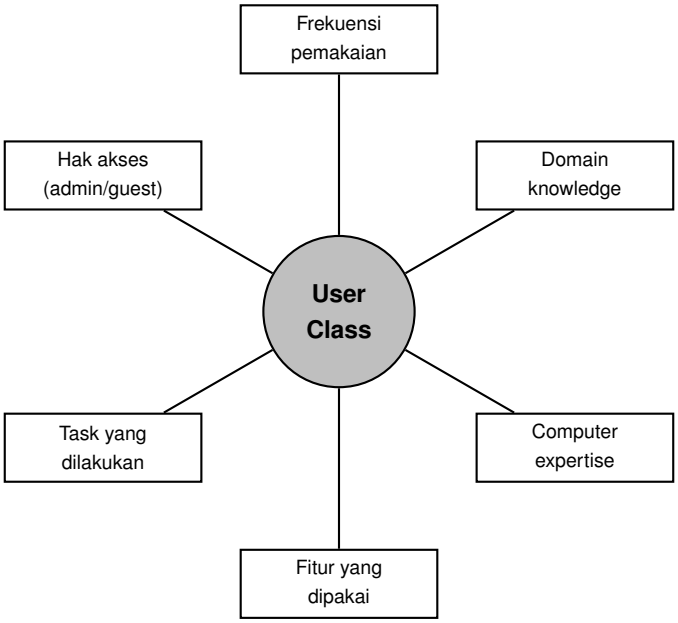


Figure 3. Enam dimensi yang membedakan user class.

Setelah mengelompokkan user ke dalam kelas, kita bisa menandai beberapa kelas sebagai **favored**. Favored user class adalah kelas yang paling kritis bagi keberhasilan produk atau bisnis. Mereka bukan yang paling banyak jumlahnya, tapi yang paling menentukan apakah produk ini berhasil atau tidak.

Contoh nyata ada di sistem chemical tracking. Chemist (sekitar 1000 orang) memakai sistem beberapa kali sehari untuk meminta bahan kimia dari vendor dan gudang. Health and Safety Department Staff memakai sistem hanya sekali per kuartal untuk menghasilkan laporan kepatuhan. Buyers (sekitar 5 orang) memproses permintaan kimia yang diajukan pihak lain. Chemical Stockroom Staff (6 teknisi dan 1 supervisor) mengelola inventori dan memproses permintaan chemist.

Kalau kita melihat dari frekuensi pemakaian, Chemist adalah yang paling sering. Tapi dari

dampak regulasi, Health and Safety Staff justru yang paling kritis. Keduanya bisa ditandai favored, meskipun alasan favored-nya berbeda.

4.2 Sumber User Requirement

Ada tujuh sumber yang bisa dipakai untuk menggali user requirement:

Pertama, wawancara dan diskusi dengan potential users. Ini sumber paling langsung, tapi juga paling memakan waktu.

Kedua, dokumen yang mendeskripsikan produk saat ini atau kompetitor. Berguna untuk memahami standar industri.

Ketiga, problem report dan enhancement request dari sistem yang sedang berjalan. Ini sumber paling jujur karena berasal dari keluhan nyata.

Keempat, analisis event dan respons dari sistem. Berguna untuk sistem yang sudah berjalan dan punya log.

Kelima, marketing survey dan kuesioner. Cocok untuk mengukur preferensi user dalam jumlah besar.

Keenam, observasi langsung terhadap user yang bekerja. Cara terbaik menemukan kebutuhan yang tidak pernah diucapkan user karena dianggap remeh.

Ketujuh, scenario analysis terhadap task user. Berguna untuk memetakan alur kerja yang kompleks.

4.3 Product Champion

Product Champion adalah beberapa member key atau influential dari user community yang merepresentasikan user dan memberi requirement. Ia menjadi interface utama antara user dari satu user class dan requirement analyst proyek. Perhatikan Gambar 4.



Figure 4. Product champion sebagai interface antara user class dan analyst.

Dua aturan penting tentang product champion.

Pertama, harus real user, bukan surrogate. Ia tidak boleh manajer yang mewakili user, atau staf pemasaran yang mengaku paham user. Ia harus benar-benar orang yang memakai sistem sehari-hari.

Kedua, harus satu per user class. Setiap user class punya perilaku dan kebutuhan berbeda. Product champion dari chemist tidak bisa mewakili buyers, meskipun keduanya bekerja di perusahaan yang sama.

Alasan kedua aturan ini sederhana: kalau champion-nya bukan user asli, ia akan menyampaikan apa yang *ia pikir* user butuhkan, bukan apa yang **user benar-benar** butuhkan. Perbedaan kecil ini, kalau dibiarkan sampai tahap implementasi, akan menghasilkan sistem yang secara teknis berjalan tapi secara praktis tidak terpakai.

5 System Requirement: Functional

Functional requirement adalah perilaku yang harus produk lakukan atau dukung. Requirement ini dipakai tim pengembangan untuk membangun perilaku sistem. Biasanya dinyatakan dengan input, output, dan deskripsi perilaku itu sendiri. Perhatikan Gambar 5.

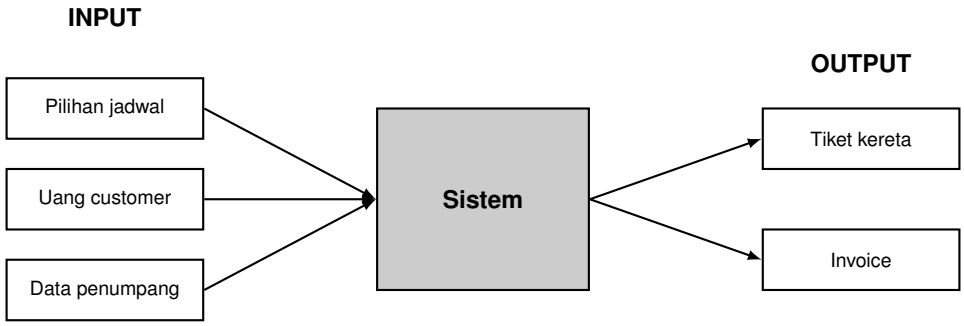


Figure 5. Struktur khas functional requirement: input, proses, output.

Contoh-contoh functional requirement yang sering muncul:

Contoh sistem, "Sistem menampilkan nama, ukuran total, ruang tersedia, dan format dari flash drive yang terhubung ke port USB."

Contoh singkat, "Tambahkan customer." "Cetak invoice."

Contoh dari produk fisik, kalau kita memandang karton susu sebagai sistem, functional requirement-nya adalah "kemampuan menampung cairan tanpa bocor."

Yang membedakan functional requirement dari yang lain adalah: ia bisa diuji dengan skenario konkret. Kalau kita tidak bisa membayangkan cara mengujinya dengan data nyata, kalimat itu bukan functional requirement, mungkin masih user requirement atau bahkan business requirement.

6 System Requirement: Non-Functional

Non-functional requirement mendeskripsikan **seberapa baik** produk harus tampil. Ia menyentuh faktor kualitas produk. Kadang disebut quality requirements. Perhatikan Gambar 6.

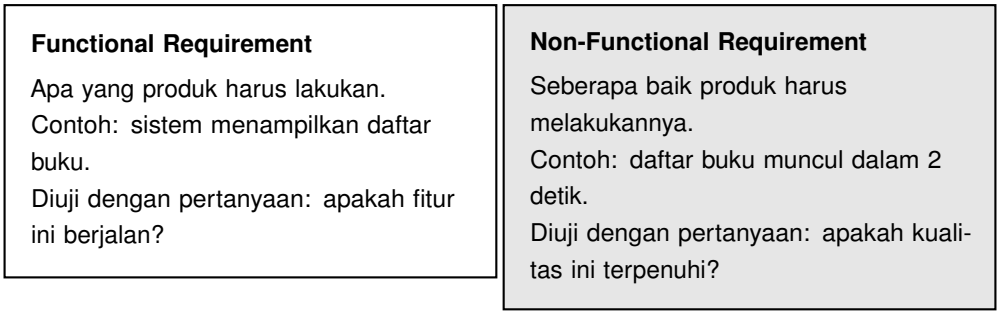


Figure 6. Perbedaan cara pandang functional dan non-functional requirement.

Kategori non-functional requirement yang lazim meliputi: performance (response time, throughput, utilization, static volumetric), scalability, maintainability, availability, security, reliability, manageability, recoverability, dan environmental.

Contoh klasik dari slide: helm proyek (hard hat) punya non-functional requirement "tidak boleh pecah pada tekanan kurang dari 10.000 PSI." Contoh ini memperjelas bahwa non-functional selalu bisa diukur dengan angka atau kriteria yang bisa diverifikasi.

Yang perlu dihindari adalah non-functional requirement yang tidak terukur, seperti "sistem harus cepat", "sistem harus aman", "sistem harus mudah digunakan". Tiga kalimat itu bukan requirement, melainkan harapan. Ubah menjadi "response time kurang dari 2 detik pada 95 persen request", "password disimpan dengan hash bcrypt", "user bisa menyelesaikan task utama dalam 3 klik atau kurang".

7 Alat Uji Klasifikasi

Saat menemukan satu kalimat di lapangan, kita kadang bingung: ini functional atau non-functional? Perhatikan Gambar 7.

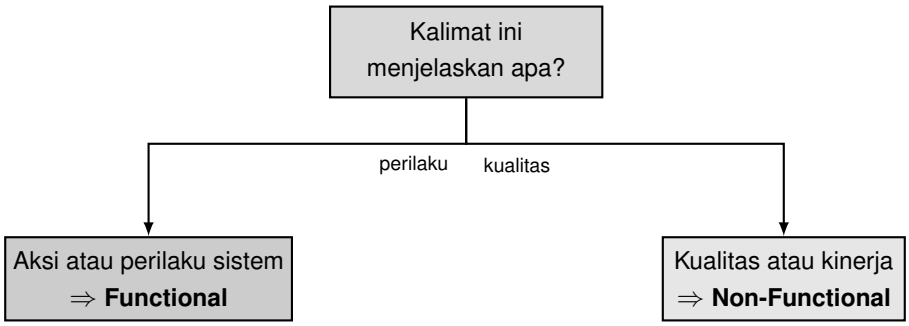


Figure 7. Alat uji cepat untuk mengklasifikasi requirement.

Aturan penggunaannya sederhana. Baca kalimat requirement yang ditemukan. Kalau ia menjelaskan **apa yang sistem harus lakukan** (misalnya "menampilkan daftar", "menyimpan riwayat", "mengirim notifikasi"), ia adalah functional requirement. Kalau ia menjelaskan **seberapa baik sistem harus melakukannya** (misalnya "dalam 2 detik", "tersedia 99 persen", "dengan enkripsi AES-256"), ia adalah non-functional requirement.

Kalau ragu, coba ubah kalimatnya menjadi pertanyaan uji. Functional requirement diuji dengan "apakah fitur ini berjalan?". Non-functional diuji dengan "apakah angka ini tercapai?". Kalau tidak ada satu pun pertanyaan yang bisa dijawab, kalimat itu masih terlalu kabur untuk masuk dokumen requirement.

8 Perbandingan Ringkasan

Tabel 1 merangkum keempat tipe requirement dalam satu pandangan.

Table 1. Perbandingan empat tipe requirement.

Tipe	Dijawab untuk	Bentuk Khas	Pembaca Utama
Business	Mengapa proyek ini ada	Vision, scope, business case	Sponsor
User	Apa yang bisa user lakukan	Use case, user story	End user
Functional	Apa yang sistem lakukan	Input, proses, output	Developer
Non-functional	Seberapa baik sistem bekerja	Angka, kriteria kualitas	Developer, tester

Setelah tabel ini, satu paragraf pengingat: keempat tipe ini tidak berdiri sendiri. Setiap functional requirement harus bisa dilacak ke satu user requirement, yang pada gilirannya harus bisa dilacak ke satu business requirement. Tanpa rantai pelacakan ini, requirement kehilangan konteks dan mudah berubah tanpa alasan yang jelas.

9 Studi Kasus: Sistem Penjadwalan Kuliah

Bagian ini memandu sepuluh langkah untuk menghasilkan artefak minggu keempat. Kasus yang dipakai: sistem penjadwalan kuliah. Tujuannya mengotomasi proses penyusunan jadwal yang sebelumnya manual oleh staf akademik.

9.1 Langkah 1: Tulis Business Requirement

Sebelum apa pun, tulis mengapa proyek ini ada.

```
Business Requirement:

Tujuan:
Mengotomasi penyusunan jadwal kuliah agar staf
akademik tidak lagi menyusun jadwal manual yang
memakan waktu 2 minggu per semester.

Ukuran keberhasilan:
- Penyusunan jadwal selesai dalam 1 hari
- Tidak ada bentrok ruang atau dosen
- Dosen bisa menyampaikan preferensi waktu

Pembaca dokumen: Kepala Staf Akademik
```

9.2 Langkah 2: Tulis User Requirement

Setelah business requirement, tulis apa yang tiap user class harus bisa lakukan.

```
User Requirement:

Staf Akademik (favored):
- Bisa mengunggah daftar mata kuliah dan dosen
- Bisa menekan tombol "Generate Jadwal"
- Bisa mengunduh jadwal dalam format PDF

Dosen (supporting):
- Bisa mengisi preferensi waktu mengajar
- Bisa melihat jadwal hasil generate

Mahasiswa (supporting):
- Bisa melihat jadwal di halaman publik
- Bisa memfilter jadwal berdasarkan semester

Pembaca dokumen: Product Champion setiap user class
```

9.3 Langkah 3: Analisis Business Process

Sebelum mengotomasi, petakan proses yang sedang berjalan.

```
Proses Tradisional (sebelum sistem):
1. Staf akademik meminta ketersediaan dosen
2. Staf mencatat preferensi dosen di Excel
3. Staf memeriksa ketersediaan ruang manual
4. Staf menyusun jadwal di Excel
5. Staf memeriksa bentrok satu per satu
6. Jika bentrok, ulangi dari langkah 3
7. Staf menerbitkan jadwal di papan pengumuman

Masalah:
- Langkah 5 bisa memakan 2 minggu
- Bentrok sering terlewat karena human error
- Revisi harus mengulang seluruh proses

Proses Otomasi (setelah sistem):
1. Staf mengunggah daftar mata kuliah dan dosen
2. Sistem menerima preferensi dosen via form
3. Sistem memeriksa bentrok otomatis
4. Sistem menghasilkan jadwal optimal
5. Staf meninjau hasil, klik "Publish"
6. Sistem menerbitkan jadwal di halaman publik
```

9.4 Langkah 4: Identifikasi Sumber User Requirement

Dari tujuh sumber yang ada, pilih minimal tiga.

```
Sumber User Requirement yang Dipakai:

1. Wawancara dengan Kepala Staf Akademik (90 menit)
   Topik: kendala penyusunan manual, harapan sistem

2. Wawancara dengan 5 dosen (masing-masing 20 menit)
   Topik: preferensi waktu, konflik yang sering dialami

3. Observasi langsung di ruang staf akademik (2 hari)
   Topik: melihat langsung alur penyusunan jadwal,
   dokumen apa yang dipakai, di mana waktu paling
   banyak terbuang

4. Analisis dokumen: kumpulkan file Excel jadwal
   3 semester terakhir, identifikasi pola bentrok
```

9.5 Langkah 5: Petakan User Class

Gunakan enam dimensi untuk mengelompokkan user.

```
User Class berdasarkan Enam Dimensi:

Staf Akademik (favored):
- Frekuensi: tiap awal semester (intens)
- Domain knowledge: tinggi
- Computer expertise: menengah
- Fitur dipakai: upload, generate, review, publish
- Task: menyusun jadwal dari nol
- Hak akses: admin

Dosen:
- Frekuensi: tiap awal semester (singkat)
- Domain knowledge: tinggi di bidangnya
- Computer expertise: bervariasi
- Fitur dipakai: isi preferensi, lihat jadwal
- Task: menyampaikan preferensi
- Hak akses: user terbatas

Mahasiswa:
- Frekuensi: tiap awal semester (baca saja)
- Domain knowledge: menengah
- Computer expertise: tinggi
- Fitur dipakai: lihat jadwal
- Task: mencari jadwal kuliah
```

```
- Hak akses: guest (baca saja)
```

9.6 Langkah 6: Rekrut Product Champion

Untuk setiap user class, tunjuk satu champion.

```
Product Champion:

Staf Akademik:
- Nama: Ibu Sri (Kepala Staf Akademik)
- Alasan: pengguna harian, paham semua kendala,
  bukan manajer yang jauh dari lapangan

Dosen:
- Nama: Pak Budi (Dosen Tetap, 10 tahun mengajar)
- Alasan: pernah mengalami konflik jadwal, aktif
  mengisi preferensi, bukan ketua program studi
  yang hanya melihat laporan

Mahasiswa:
- Nama: Sari (Mahasiswa Semester 5)
- Alasan: aktif mencari jadwal, pernah komplain
  soal jadwal bentrok, bukan ketua BEM yang sibuk
  dengan kegiatan lain
```

9.7 Langkah 7: Tulis Functional Requirement

Dari user requirement, turunkan menjadi perilaku sistem.

```
Functional Requirement:

FR-01: Staf akademik mengunggah file CSV berisi
daftar mata kuliah dan dosen.
Input: file CSV dengan kolom kode, nama, SKS,
dosen_pengampu.
Output: konfirmasi jumlah mata kuliah yang berhasil
diunggah.

FR-02: Sistem memeriksa bentrok ruang dan dosen
sebelum menghasilkan jadwal.
Input: data mata kuliah, data dosen, data ruang,
data preferensi.
Output: laporan bentrok (jika ada) atau jadwal
final (jika tidak ada bentrok).

FR-03: Staf akademik menerbitkan jadwal ke halaman
```

```
publik.
Input: jadwal yang sudah disetujui.
Output: URL halaman publik yang bisa diakses
tanpa login.

Tracing:
- FR-01 melacak ke user requirement Staf Akademik
- FR-02 melacak ke user requirement Staf Akademik
- FR-03 melacak ke user requirement Mahasiswa
```

9.8 Langkah 8: Tulis Non-Functional Requirement

Untuk setiap functional requirement, pikirkan seberapa baik ia harus bekerja.

```
Non-Functional Requirement:

NFR-01 (Performance): Proses generate jadwal untuk
200 mata kuliah dan 50 dosen selesai dalam waktu
kurang dari 5 menit.

NFR-02 (Availability): Halaman jadwal publik
tersedia 99,5 persen waktu per bulan, di luar
jadwal pemeliharaan terjadwal.

NFR-03 (Security): Hanya staf akademik dengan
akun admin yang bisa menerbitkan jadwal.
Percobaan publish dari akun non-admin ditolak
dengan log audit.

NFR-04 (Scalability): Sistem bisa menangani
penambahan hingga 500 mata kuliah tanpa perubahan
arsitektur.

Tracing:
- NFR-01 terkait FR-02
- NFR-02 terkait FR-03
- NFR-03 terkait FR-03
- NFR-04 terkait FR-01 dan FR-02
```

9.9 Langkah 9: Uji Klasifikasi dan Traceability

Untuk setiap requirement yang sudah ditulis, jalankan dua uji.

```
Uji Klasifikasi:

Kalimat: "Sistem memeriksa bentrok ruang dan dosen
```

```
sebelum menghasilkan jadwal."
Uji: menjelaskan perilaku (apa yang sistem lakukan)?
Hasil: ya, ini Functional.

Kalimat: "Proses generate selesai dalam 5 menit."
Uji: menjelaskan kualitas (seberapa baik)?
Hasil: ya, ini Non-Functional.

Uji Traceability:

Setiap FR punya user requirement di atasnya?
FR-01 ya, punya.
FR-02 ya, punya.
FR-03 ya, punya.

Setiap NFR punya FR yang didukung?
NFR-01 ya, mendukung FR-02.
NFR-02 ya, mendukung FR-03.
NFR-03 ya, mendukung FR-03.
NFR-04 ya, mendukung FR-01 dan FR-02.

Requirement tanpa traceability: tidak ada.
```

9.10 Langkah 10: Validasi Silang

Sebelum dokumen diserahkan, validasi ke tiap pembaca.

```
Agenda Validasi:

Sesi 1 - dengan Sponsor (30 menit):
- Konfirmasi business requirement dan ukuran
  keberhasilan

Sesi 2 - dengan Product Champion Staf Akademik
(45 menit):
- Uji user requirement dan functional requirement

Sesi 3 - dengan Product Champion Dosen (30 menit):
- Uji user requirement, terutama soal preferensi
  waktu

Sesi 4 - dengan Product Champion Mahasiswa
(20 menit):
- Uji user requirement, terutama soal akses publik

Kriteria validasi:
- Semua champion setuju isi dokumen
- Tidak ada requirement tanpa pembaca yang jelas
```

```
- Tidak ada requirement tanpa traceability
```

Sepuluh langkah di atas menghasilkan empat artefak siap kumpul: dokumen business requirement dan business process, dokumen user requirement beserta peta user class, dokumen system requirement beserta traceability matrix, dan catatan validasi silang. Inilah deliverable minggu keempat.

10 Kesalahan Umum

Tabel 2 merangkum kesalahan yang paling sering muncul saat membedakan tipe requirement.

Table 2. Kesalahan umum pada pekerjaan tipe requirement minggu keempat.

Kesalahan	Akibat	Perbaikan
Menulis business requirement dalam bahasa developer	Sponsor tidak bisa membaca, developer salah paham	Tulis business requirement dalam bahasa bisnis
Menulis user requirement tanpa user class	Kebutuhan satu kelas terabaikan	Definisikan user class pakai enam dimensi
Menganggap semua user identik	Fitur satu user mengganggu user lain	Tandai favored, desain per kelas
Menunjuk manajer sebagai product champion	Champion menyampaikan asumsi, bukan kebutuhan	Wajib real user, bukan surrogate
Menulis functional requirement tanpa input-output	Tidak bisa diuji	Selalu sertakan input, proses, output
Menulis non-functional tanpa angka	Tidak bisa diverifikasi	Ganti “cepat” dengan “kurang dari 2 detik”
Mencampur functional dan non-functional dalam satu kalimat	Sulit dilacak	Pisahkan jadi dua kalimat
Tidak punya traceability antar level	Requirement tanpa alasan jelas	Setiap FR melacak ke UR, setiap UR ke BR
Melewatkan business process analysis	Otomasi salah proses	Petakan proses lama sebelum mengotomasi

Tiga kesalahan yang paling sering terjadi adalah **menulis non-functional requirement tanpa angka**, **tidak memetakan user class sebelum menulis user requirement**, dan **tidak punya traceability antar level**. Ketiganya sama-sama membuat dokumen requirement kehilangan satu sifat penting: bisa diperiksa kebenarannya. Dokumen yang tidak bisa diperiksa adalah dokumen yang tidak akan pernah disepakati.

11 Lembar Rujukan Cepat

Tabel 3 merangkum istilah dan template yang wajib dikuasai sebelum pertemuan kelima.

Table 3. Lembar rujukan cepat untuk minggu keempat.

Kebutuhan atau Istilah	Definisi atau Template
Business requirement	Mengapa proyek ada, apa yang ingin dicapai, visi dan scope
User requirement	Apa yang user bisa lakukan dengan produk
Functional requirement	Perilaku sistem, dinyatakan dengan input, proses, output
Non-functional requirement	Kualitas sistem, harus terukur dengan angka atau kriteria
User class	Kelompok user berdasarkan enam dimensi
Favored user class	Kelas user paling kritis bagi keberhasilan produk
Product champion	Real user dari satu user class yang menjadi interface ke analyst
Sumber user requirement	Wawancara, dokumen, problem report, event, survey, observasi, scenario
Business process analysis	Memetakan proses lama sebelum mengotomasi
Traceability	Setiap FR melacak ke UR, setiap UR melacak ke BR
Template user story	“Sebagai [role], saya ingin [goal], agar [benefit].”
Template functional requirement	“Sistem [aksi]. Input: [...]. Output: [...].”
Template non-functional	“Sistem [kualitas] minimal/ maksimal [angka] pada [kondisi].”
Enam dimensi user class	Frekuensi, domain knowledge, computer expertise, fitur, task, hak akses
Hindari	“cepat”, “aman”, “mudah” tanpa angka atau kriteria

12 Kesimpulan

Requirement hidup di empat level: business, user, functional, dan non-functional. Masing-masing punya pembaca utama berbeda dan diuji dengan cara berbeda. Tiga prinsip penutup:

1. **Bedakan level sebelum menulis.** Setiap kalimat requirement harus jelas levelnya. Business requirement untuk sponsor, user requirement untuk end user, functional dan non-functional untuk developer. Mencampur level dalam satu kalimat adalah sumber kebingungan yang paling sering muncul di revisi dokumen.

2. **Uji dengan pertanyaan, bukan perasaan.** Functional requirement diuji dengan “apakah fitur berjalan?”. Non-functional diuji dengan “apakah angka tercapai?”. Kalau tidak ada pertanyaan yang bisa dijawab, kalimat itu masih terlalu kabur untuk masuk dokumen.

3. **Lacak ke atas, lacak ke bawah.** Setiap functional requirement wajib melacak ke user requirement di atasnya. Setiap user requirement wajib melacak ke business requirement. Tanpa traceability, requirement kehilangan konteks dan mudah berubah tanpa alasan yang jelas.

Pada pertemuan berikutnya, prinsip-prinsip ini akan dipakai untuk mempelajari vision dan scope secara lebih mendalam. Lembar rujukan cepat di atas adalah bekal minimum yang harus dibawa.

Referensi

References

[1] Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson.

[2] Wiegers, K., & Beatty, J. (2013). *Software Requirements* (3rd ed.). Microsoft Press.

[3] Robertson, S., & Robertson, J. (2012). *Mastering the Requirements Process: Getting Requirements Right* (3rd ed.). Addison-Wesley.

[4] International Institute of Business Analysis. (2015). *A Guide to the Business Analysis Body of Knowledge (BABOK Guide)* (v3.0). IIBA.

[5] Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill Education.

[6] IEEE. (1998). *IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998)*. Institute of Electrical and Electronics Engineers.

[7] Cockburn, A. (2000). *Writing Effective Use Cases*. Addison-Wesley.