Dekumen Rekayasa Kebutuhan Perangkat Lunak Aplikasi Mobile Bengkelin PT Otomotif Digital Indonesia

Bagian A :

A. Identifikasi Stakeholder dan Teknik Elisitasi

1. Pemilik Produk (Product Owner) / Manajemen PT Otomotif Digital Indonesia

- Peran: Pihak yang bertanggung jawab atas visi produk, prioritas fitur, dan keselarasan aplikasi dengan tujuan bisnis perusahaan. Mereka memiliki pemahaman mendalam tentang pasar, target pengguna, dan tujuan strategis Bengkelin.
- Teknik Elisitasi yang Cocok:
    - Wawancara Terstruktur (Structured Interview): Melakukan serangkaian pertanyaan yang telah disiapkan secara spesifik untuk memahami visi bisnis, tujuan MVP, kriteria sukses, batasan anggaran, dan harapan high-level terhadap aplikasi.
    - Alasan: Wawancara terstruktur memastikan semua area krusial tercover dan memungkinkan penggalian informasi mendalam mengenai strategi bisnis dan prioritas fitur dari sudut pandang manajemen.

2. Calon Pengguna Aplikasi (Vehicle Owners)

- Peran: Individu yang akan menggunakan aplikasi Bengkelin untuk mencari bengkel dan memesan layanan. Masukan dari mereka sangat penting untuk memastikan aplikasi mudah digunakan, relevan, dan memenuhi kebutuhan nyata.
- Teknik Elisitasi yang Cocok:
    - Survei (Surveys): Menyebarkan kuesioner daring kepada demografi target untuk mengumpulkan data kuantitatif mengenai preferensi, kebiasaan servis kendaraan, fitur yang diinginkan, dan masalah yang sering dihadapi.
    - Fokus Grup (Focus Groups): Mengumpulkan sekelompok kecil calon pengguna untuk berdiskusi secara mendalam tentang ide-ide fitur, pain points, dan user experience yang diharapkan.
    - Alasan: Survei efektif untuk mendapatkan gambaran umum dari banyak responden, sedangkan fokus grup memungkinkan eksplorasi

mendalam dan pemahaman nuansa dari berbagai perspektif pengguna, membantu mengidentifikasi kebutuhan yang mungkin tidak terpikirkan.

3. Pemilik Bengkel Rekanan

- Peran: Pihak yang menyediakan layanan servis melalui aplikasi Bengkelin. Masukan mereka krusial untuk memastikan integrasi yang mulus, pengelolaan pemesanan yang efisien, dan fitur yang mendukung operasional bengkel.
- Teknik Elisitasi yang Cocok:
    - Wawancara Semi-Terstruktur (Semi-Structured Interview): Melakukan wawancara dengan panduan pertanyaan, namun tetap fleksibel untuk mengeksplorasi isu-isu yang muncul selama percakapan. Ini akan fokus pada proses bisnis bengkel, bagaimana mereka mengelola pesanan saat ini, dan harapan mereka terhadap sistem pemesanan online.
    - Observasi (Observation): Mengunjungi beberapa bengkel rekanan untuk mengamati langsung alur kerja mereka, bagaimana mereka menerima dan memproses pesanan, serta tantangan yang mereka hadapi dalam operasional sehari-hari.
    - Alasan: Wawancara semi-terstruktur memungkinkan fleksibilitas untuk menggali detail spesifik dari berbagai jenis bengkel. Observasi memberikan pemahaman kontekstual tentang lingkungan kerja dan proses aktual, yang membantu merancang fitur yang praktis dan efisien untuk pihak bengkel.

4. Tim Pengembang / Pengembang Aplikasi (Development Team)

- Peran: Individu yang bertanggung jawab untuk merancang, membangun, dan menguji aplikasi Bengkelin. Mereka perlu memahami persyaratan fungsional dan non-fungsional untuk memastikan kelayakan teknis dan efisiensi pengembangan.
- Teknik Elisitasi yang Cocok:
    - Sesi Brainstorming (Brainstorming Sessions): Mengadakan pertemuan kolaboratif untuk menghasilkan ide-ide solusi teknis, mengidentifikasi potensi tantangan implementasi, dan membahas arsitektur sistem.
    - Workshop Persyaratan (Requirements Workshops): Melakukan sesi interaktif di mana stakeholder lain (seperti Product Owner) berpartisipasi untuk memperjelas persyaratan, membuat diagram alir, dan menyepakati definisi fitur.
    - Alasan: Sesi brainstorming mendorong kreativitas dan menemukan solusi teknis terbaik dari internal tim. Workshop persyaratan memastikan pemahaman yang sama antara tim pengembang dan

stakeholder bisnis, mengurangi miskomunikasi dan mempercepat proses definisi fitur.

Bagian B :

B. Analisis Kebutuhan

Analisis kebutuhan ini menguraikan fitur spesifik yang harus dimiliki aplikasi (kebutuhan fungsional) dan kualitas serta batasan yang harus dipenuhi aplikasi (kebutuhan non-fungsional).

1. Kebutuhan Fungsional

Kebutuhan fungsional mendefinisikan apa yang harus dilakukan sistem. Kita akan menggunakan Use Case Diagram untuk menggambarkan interaksi utama pengguna dengan sistem dan User Story untuk menjelaskan detail fitur dari sudut pandang pengguna.

a. Use Case Diagram (Gambaran Umum)

```
graph TD
    A[Pengguna] --> B(Registrasi Akun)
    A --> C(Login)
    A --> D(Cari Bengkel)
    A --> E(Lihat Detail Bengkel)
    A --> F(Pesan Layanan)
    A --> G(Lihat Riwayat Servis)
    A --> H(Beri Rating & Review)

    I[Bengkel Rekanan] --> J(Kelola Profil Bengkel)
    I --> K(Terima/Tolak Pesanan)
    I --> L(Perbarui Status Servis)

    M[Admin Sistem] --> N(Verifikasi Bengkel)
    M --> O(Kelola Pengguna)
    M --> P(Kelola Data Master)
```

b. User Stories (Detail Fitur)

Berikut adalah user stories yang merinci kebutuhan fungsional:

Modul Pengguna:

- Registrasi & Login:
    - Sebagai pengguna baru, saya ingin mendaftar akun menggunakan email/nomor telepon dan kata sandi, sehingga saya bisa mengakses fitur-fitur aplikasi.
    - Sebagai pengguna terdaftar, saya ingin masuk ke akun saya, sehingga saya bisa melanjutkan penggunaan aplikasi.
    - Sebagai pengguna, saya ingin reset kata sandi jika saya lupa, sehingga saya bisa kembali mengakses akun saya.
- Pencarian & Penemuan Bengkel:
    - Sebagai pengguna, saya ingin mencari bengkel berdasarkan lokasi saya saat ini, sehingga saya bisa menemukan bengkel terdekat.
    - Sebagai pengguna, saya ingin mencari bengkel berdasarkan jenis kendaraan atau layanan spesifik, sehingga saya bisa menemukan bengkel yang sesuai dengan kebutuhan saya.
    - Sebagai pengguna, saya ingin melihat daftar bengkel yang direkomendasikan (misalnya, berdasarkan rating tertinggi atau popularitas), sehingga saya bisa memilih bengkel dengan reputasi baik.
    - Sebagai pengguna, saya ingin melihat detail profil bengkel (alamat, jam operasional, layanan yang tersedia, harga, rating, ulasan), sehingga saya bisa membuat keputusan yang terinformasi.
- Pemesanan Layanan (Booking):
    - Sebagai pengguna, saya ingin memilih layanan servis yang tersedia di bengkel, sehingga saya bisa memesan sesuai kebutuhan.
    - Sebagai pengguna, saya ingin memilih jadwal servis (tanggal dan waktu) yang tersedia, sehingga saya bisa menyesuaikan dengan ketersediaan saya.
    - Sebagai pengguna, saya ingin mengisi detail kendaraan (merk, model, tahun, plat nomor) saat memesan, sehingga bengkel memiliki informasi awal.
    - Sebagai pengguna, saya ingin mengkonfirmasi pemesanan setelah meninjau detailnya, sehingga pemesanan saya tercatat.
    - Sebagai pengguna, saya ingin menerima notifikasi konfirmasi pemesanan, sehingga saya yakin pesanan saya berhasil.
- Riwayat Servis:
    - Sebagai pengguna, saya ingin melihat riwayat semua servis kendaraan saya, sehingga saya bisa melacak pemeliharaan kendaraan saya.

    - Sebagai pengguna, saya ingin melihat detail setiap riwayat servis (layanan yang dilakukan, tanggal, biaya, bengkel), sehingga saya memiliki catatan lengkap.

- Rating & Review:
    - Sebagai pengguna, saya ingin memberikan rating (bintang) setelah servis selesai, sehingga saya bisa memberikan umpan balik cepat.
    - Sebagai pengguna, saya ingin menulis ulasan (review) tentang pengalaman servis saya, sehingga pengguna lain bisa mendapatkan informasi dan bengkel bisa meningkatkan layanannya.

Modul Bengkel Rekanan:

- Manajemen Profil Bengkel:
    - Sebagai pemilik bengkel, saya ingin mengelola informasi profil bengkel saya (jam operasional, alamat, layanan, harga), sehingga informasi yang ditampilkan kepada pengguna selalu akurat.
    - Sebagai pemilik bengkel, saya ingin melihat dan mengelola ketersediaan jadwal servis saya, sehingga saya bisa mengatur kapasitas bengkel.

- Manajemen Pemesanan:
    - Sebagai pemilik bengkel, saya ingin menerima notifikasi ketika ada pesanan baru, sehingga saya bisa segera merespons.
    - Sebagai pemilik bengkel, saya ingin melihat detail pesanan yang masuk, sehingga saya bisa mempersiapkan diri.
    - Sebagai pemilik bengkel, saya ingin menerima atau menolak pesanan, sehingga saya bisa mengelola kapasitas dan ketersediaan.
    - Sebagai pemilik bengkel, saya ingin memperbarui status servis (misalnya, "Dalam Pengerjaan", "Selesai"), sehingga pengguna dapat memantau progresnya.

Modul Admin Sistem (Back-end):

- Sebagai admin sistem, saya ingin memverifikasi dan mengaktifkan akun bengkel rekanan, sehingga hanya bengkel terpercaya yang terdaftar di aplikasi.
- Sebagai admin sistem, saya ingin mengelola data pengguna (melihat, menonaktifkan akun), sehingga saya dapat menjaga integritas sistem.
- Sebagai admin sistem, saya ingin mengelola data master (jenis kendaraan, jenis layanan), sehingga data di aplikasi selalu terstandarisasi dan terkini.

2. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional mendefinisikan kriteria untuk mengevaluasi operasi sistem, bukan perilaku spesifik. Ini penting untuk kualitas dan keberhasilan jangka panjang aplikasi.

- Performa (Performance):
    - Aplikasi harus memiliki waktu respons maksimal 3 detik untuk sebagian besar tindakan pengguna (pencarian, booking).
    - Aplikasi harus mampu menangani minimal 1.000 transaksi pemesanan per hari tanpa penurunan performa yang signifikan.
    - Pencarian bengkel berdasarkan lokasi harus cepat dan akurat.
- Keandalan (Reliability):
    - Sistem harus memiliki ketersediaan (uptime) minimal 99,5% setiap bulan.
    - Data pemesanan harus tersimpan dengan aman dan tidak boleh hilang dalam situasi apapun (misalnya, kegagalan server).
    - Sistem harus dapat pulih dari kegagalan dalam waktu maksimal 1 jam.
- Skalabilitas (Scalability):
    - Sistem harus mampu mendukung penambahan jumlah pengguna dan bengkel rekanan yang terus bertambah tanpa perlu perubahan arsitektur mayor.
    - Sistem harus dapat mengakomodasi peningkatan volume data riwayat servis dan rating/review di masa depan.
- Keamanan (Security):
    - Semua data pengguna (informasi pribadi, riwayat servis) harus dienkripsi saat transit dan saat disimpan (at rest).
    - Harus ada mekanisme autentikasi dan otorisasi yang kuat untuk pengguna dan bengkel.
    - Sistem harus terlindungi dari serangan siber umum seperti SQL injection dan XSS.
- Kemudahan Penggunaan (Usability):
    - Antarmuka pengguna (UI) harus intuitif dan mudah dipahami oleh berbagai kalangan pengguna.

    - Proses registrasi dan pemesanan harus minimalis dan mudah diselesaikan.
    - Desain aplikasi harus konsisten di seluruh fitur.

- Kompatibilitas (Compatibility):

    - Aplikasi mobile harus kompatibel dengan sistem operasi Android (versi terbaru dan 2 versi sebelumnya) dan iOS (versi terbaru dan 2 versi sebelumnya).
    - Aplikasi harus berfungsi dengan baik di berbagai ukuran layar perangkat mobile.

- Maintainability (Kemudahan Pemeliharaan):

    - Kode harus terstruktur dengan baik dan terdokumentasi, sehingga mudah untuk diperbaiki dan diperbarui di masa mendatang.
    - Sistem harus dirancang dengan modularitas tinggi untuk memudahkan penambahan fitur baru.

- Portabilitas (Portability):

    - Sistem harus dirancang sedemikian rupa sehingga, jika diperlukan, dapat di-deploy di lingkungan cloud provider yang berbeda dengan upaya minimal.

Bagian C :

C. Ringkasan Software Requirements Specification (SRS) - Bengkelin

1. Pendahuluan

Dokumen ini menjelaskan persyaratan perangkat lunak untuk aplikasi mobile "Bengkelin", yang akan dikembangkan oleh PT Otomotif Digital Indonesia. Aplikasi ini bertujuan untuk mempermudah pengguna kendaraan dalam memesan layanan servis kendaraan secara online dari berbagai bengkel rekanan yang sudah terverifikasi. Dokumen ini akan menjadi panduan utama bagi tim pengembangan dalam merancang, membangun, dan menguji aplikasi Bengkelin, serta sebagai referensi bagi stakeholder lainnya.

2. Tujuan Sistem

Tujuan utama dari pengembangan aplikasi Bengkelin adalah:

- Mempermudah Pengguna: Menyediakan platform yang intuitif dan efisien bagi pemilik kendaraan untuk menemukan dan memesan layanan servis dari bengkel terverifikasi.

- Meningkatkan Aksesibilitas Bengkel: Memperluas jangkauan pasar bagi bengkel rekanan dan mengoptimalkan manajemen pesanan mereka melalui platform digital.
- Membangun Ekosistem Terpercaya: Menghadirkan daftar bengkel yang terverifikasi dan sistem rating & review untuk membangun kepercayaan pengguna terhadap layanan yang diberikan.
- Efisiensi Waktu: Mengurangi waktu dan upaya yang diperlukan pengguna dalam mencari dan memesan layanan servis kendaraan secara manual.
- Digitalisasi Proses: Mendorong digitalisasi proses pemesanan dan manajemen servis kendaraan di Indonesia.

3. Lingkup Sistem

Aplikasi Bengkelin akan beroperasi sebagai platform penghubung antara pemilik kendaraan (pengguna) dan bengkel rekanan. Lingkup sistem mencakup:

- Aplikasi Mobile Pengguna: Antarmuka utama bagi pemilik kendaraan untuk berinteraksi dengan sistem.
- Aplikasi/Panel Bengkel Rekanan: Antarmuka untuk bengkel dalam mengelola profil, layanan, jadwal, dan pesanan.
- Sistem Backend (Admin): Sistem manajemen internal untuk PT Otomotif Digital Indonesia dalam mengelola pengguna, bengkel, verifikasi, dan data master.

Fitur-fitur utama yang akan dicakup dalam MVP (Minimum Viable Product) ini meliputi:

- Registrasi dan otentikasi pengguna (login/logout).
- Pencarian bengkel berdasarkan lokasi dan filter dasar (jenis layanan).
- Melihat detail profil bengkel (informasi kontak, jam buka, layanan, rating).
- Proses pemesanan layanan servis (pemilihan layanan, tanggal, waktu).
- Notifikasi status pemesanan (konfirmasi, diterima/ditolak).
- Melihat riwayat servis yang telah dilakukan.
- Memberikan rating dan ulasan terhadap bengkel.
- Manajemen profil dan layanan oleh bengkel rekanan.
- Penerimaan/penolakan pesanan oleh bengkel.
- Update status servis oleh bengkel.
- Verifikasi bengkel dan manajemen pengguna oleh admin sistem.

Fitur-fitur yang tidak termasuk dalam lingkup MVP ini namun mungkin dipertimbangkan di fase berikutnya:

- Sistem pembayaran terintegrasi.
- Chat/pesan instan antara pengguna dan bengkel.

- Fitur promo dan diskon.
- Integrasi dengan sistem kalender eksternal.
- Fitur GPS navigasi ke bengkel.

4. Daftar Kebutuhan Sistem

Berikut adalah daftar kebutuhan fungsional dan non-fungsional untuk aplikasi Bengkelin:

4.1. Kebutuhan Fungsional (Functional Requirements)

- FR-1: Manajemen Akun Pengguna
    - FR-1.1: Pengguna dapat melakukan registrasi akun baru (email/nomor telepon, kata sandi).
    - FR-1.2: Pengguna dapat masuk (login) ke akun yang sudah terdaftar.
    - FR-1.3: Pengguna dapat melakukan reset kata sandi.
- FR-2: Pencarian dan Informasi Bengkel
    - FR-2.1: Pengguna dapat mencari bengkel berdasarkan lokasi terdekat.
    - FR-2.2: Pengguna dapat mencari bengkel berdasarkan jenis kendaraan atau layanan.
    - FR-2.3: Pengguna dapat melihat daftar bengkel beserta informasi singkat (nama, rating, jarak).
    - FR-2.4: Pengguna dapat melihat detail profil lengkap bengkel (alamat, jam operasional, daftar layanan, harga, ulasan, rating).
- FR-3: Pemesanan Layanan Servis
    - FR-3.1: Pengguna dapat memilih layanan servis yang tersedia dari bengkel.
    - FR-3.2: Pengguna dapat memilih tanggal dan slot waktu yang tersedia untuk servis.
    - FR-3.3: Pengguna dapat memasukkan detail kendaraan untuk servis.
    - FR-3.4: Pengguna dapat mengkonfirmasi pesanan setelah meninjau detail.
    - FR-3.5: Pengguna menerima notifikasi konfirmasi pesanan (di dalam aplikasi dan/atau push notification).
- FR-4: Riwayat Servis
    - FR-4.1: Pengguna dapat melihat daftar riwayat semua servis yang pernah dilakukan.

    - FR-4.2: Pengguna dapat melihat detail setiap riwayat servis (layanan, tanggal, bengkel, status).

- FR-5: Rating dan Ulasan
    - FR-5.1: Pengguna dapat memberikan rating (bintang) setelah servis selesai.
    - FR-5.2: Pengguna dapat menulis ulasan (teks) tentang pengalaman servis.
- FR-6: Manajemen Bengkel Rekanan
    - FR-6.1: Bengkel dapat mengelola informasi profil (alamat, kontak, jam operasional, daftar layanan, harga).
    - FR-6.2: Bengkel dapat melihat daftar pesanan yang masuk.
    - FR-6.3: Bengkel dapat menerima atau menolak pesanan.
    - FR-6.4: Bengkel dapat memperbarui status servis (misalnya, "Menunggu Konfirmasi", "Diterima", "Dalam Pengerjaan", "Selesai").
    - FR-6.5: Bengkel menerima notifikasi pesanan baru.
- FR-7: Administrasi Sistem
    - FR-7.1: Admin dapat memverifikasi dan mengaktifkan/menonaktifkan akun bengkel.
    - FR-7.2: Admin dapat mengelola data master (jenis kendaraan, kategori layanan).
    - FR-7.3: Admin dapat melihat dan mengelola daftar pengguna.

4.2. Kebutuhan Non-Fungsional (Non-Functional Requirements)

- NFR-1: Performa
    - NFR-1.1: Waktu respons aplikasi maksimal 3 detik untuk sebagian besar interaksi pengguna (pencarian, pemesanan).
    - NFR-1.2: Sistem mampu menangani 1.000+ transaksi pemesanan per hari.
- NFR-2: Keandalan
    - NFR-2.1: Ketersediaan sistem (uptime) minimal 99.5% per bulan.
    - NFR-2.2: Data pengguna dan pemesanan harus tersimpan dengan aman dan memiliki mekanisme backup.
- NFR-3: Skalabilitas

    - NFR-3.1: Sistem harus mampu mengakomodasi penambahan jumlah pengguna dan bengkel hingga 10x lipat dari target awal MVP tanpa re-architecture mayor.

- NFR-4: Keamanan
    - NFR-4.1: Autentikasi pengguna dan bengkel harus kuat (misalnya, password hashing).
    - NFR-4.2: Data sensitif (misalnya, informasi pribadi) harus dienkripsi saat transmisi (SSL/TLS) dan penyimpanan.
    - NFR-4.3: Sistem harus kebal terhadap kerentanan keamanan umum (misalnya, injeksi SQL, XSS).

- NFR-5: Kemudahan Penggunaan (Usability)
    - NFR-5.1: Antarmuka pengguna (UI) harus intuitif, bersih, dan mudah dinavigasi.
    - NFR-5.2: Proses pemesanan harus dapat diselesaikan dalam maksimal 5 langkah.
    - NFR-5.3: Teks dan instruksi di aplikasi harus jelas dan mudah dipahami.

- NFR-6: Kompatibilitas
    - NFR-6.1: Aplikasi mobile harus kompatibel dengan Android versi 10 ke atas dan iOS versi 14 ke atas.
    - NFR-6.2: Aplikasi harus dapat berjalan optimal di berbagai ukuran layar perangkat mobile.

- NFR-7: Kemudahan Pemeliharaan (Maintainability)
    - NFR-7.1: Kode sumber harus terstruktur dan terdokumentasi dengan baik.
    - NFR-7.2: Sistem harus dirancang secara modular untuk memudahkan pengembangan dan perbaikan di masa mendatang.

- NFR-8: Portabilitas
    - NFR-8.1: Arsitektur sistem harus memungkinkan migrasi ke penyedia cloud lain dengan upaya minimal di masa depan.

Bagian D :

D. Validasi dan Verifikasi Kebutuhan

Untuk memastikan bahwa kebutuhan yang telah disusun untuk aplikasi Bengkelin benar, lengkap, konsisten, dan disepakati oleh semua stakeholder, kita akan menerapkan kombinasi metode validasi dan verifikasi berikut:

Metode Validasi

Validasi berfokus pada pembangunan produk yang "tepat" (apakah kita membangun apa yang benar-benar dibutuhkan stakeholder).

1. Prototyping (Low-Fidelity & High-Fidelity)

- Deskripsi: Membuat versi awal (prototipe) dari antarmuka pengguna dan alur kerja aplikasi.
    - Low-Fidelity Prototype: Sketsa kasar di kertas atau wireframes digital sederhana yang menunjukkan tata letak dasar dan navigasi. Berguna di awal proyek untuk mendapatkan feedback cepat.
    - High-Fidelity Prototype: Mockup interaktif yang lebih detail dan menyerupai tampilan akhir aplikasi, dibuat dengan tool seperti Figma atau Adobe XD.
- Penerapan: Prototipe akan ditunjukkan kepada calon pengguna dan pemilik produk/manajemen untuk mendapatkan feedback langsung tentang kemudahan penggunaan, alur kerja, dan apakah fitur-fitur yang diusulkan sesuai dengan harapan mereka.
- Alasan Pemilihan: Memungkinkan stakeholder untuk "melihat" dan "berinteraksi" dengan ide aplikasi sebelum pengembangan dimulai, sehingga dapat mengidentifikasi kesenjangan atau area yang perlu perbaikan di awal. Ini sangat efektif untuk kebutuhan user experience dan user interface.

2. Review (Walkthroughs & Inspeksi)

- Deskripsi: Sesi formal di mana dokumen persyaratan (SRS, user stories) ditinjau bersama oleh tim proyek dan stakeholder kunci.
    - Walkthroughs: Penjelasan step-by-step tentang alur fungsionalitas dan fitur kepada stakeholder.
    - Inspeksi: Tinjauan yang lebih formal dan sistematis oleh tim ahli (misalnya, tim QA, analis lain) untuk menemukan inkonsistensi, ambiguitas, atau kelalaian.
- Penerapan: Melibatkan pemilik produk, perwakilan bengkel, dan tim pengembangan untuk meninjau user stories, use case diagrams, dan daftar kebutuhan fungsional/non-fungsional.
- Alasan Pemilihan: Memastikan semua stakeholder memiliki pemahaman yang sama tentang persyaratan, mengidentifikasi ambiguitas, dan mendapatkan persetujuan resmi atas dokumen kebutuhan.

Metode Verifikasi

Verifikasi berfokus pada pembangunan produk secara "benar" (apakah kita membangun produk sesuai dengan spesifikasi). Meskipun lebih banyak dilakukan di fase pengembangan, verifikasi kebutuhan juga penting di tahap awal untuk memastikan kualitas dokumen itu sendiri.

1. Checklist dan Traceability Matrix

- Deskripsi: Menggunakan daftar periksa untuk memastikan semua komponen standar dokumen SRS terpenuhi. Traceability matrix menghubungkan setiap kebutuhan dengan sumbernya (stakeholder), desain, kode, dan test case.
- Penerapan: Analis sistem akan menggunakan checklist untuk meninjau kelengkapan dan konsistensi SRS. Traceability matrix akan dibangun untuk melacak setiap user story hingga test case yang sesuai.
- Alasan Pemilihan: Memastikan kelengkapan dokumen, konsistensi antarbagian, dan bahwa setiap kebutuhan dapat dilacak dari awal hingga akhir siklus pengembangan.

2. Model Checking / Formal Methods (untuk Kebutuhan Kritis)

- Deskripsi: Menggunakan teknik berbasis model atau matematika untuk memverifikasi properti tertentu dari sistem, terutama untuk kebutuhan yang sangat kritis. Meskipun kurang umum untuk aplikasi MVP, bisa relevan untuk aspek tertentu.
- Penerapan: Mungkin diterapkan pada alur pemesanan yang kritis untuk memastikan tidak ada kondisi deadlock atau race condition dalam manajemen slot waktu.
- Alasan Pemilihan: Meskipun cenderung kompleks, metode ini memberikan jaminan tingkat tinggi terhadap konsistensi dan kelengkapan logika untuk bagian sistem yang sangat penting.

Bagian E

E. Manajemen Perubahan Kebutuhan

Dalam pengembangan software dengan pendekatan Agile, perubahan adalah hal yang tak terhindarkan dan seringkali bahkan disambut baik sebagai bentuk adaptasi terhadap kebutuhan pasar yang berkembang. Kunci utamanya adalah bagaimana kita mengelola perubahan tersebut secara terstruktur dan transparan.

Mari kita simulasikan skenario perubahan:

Skenario Perubahan Kebutuhan: Penambahan Fitur Pembayaran Digital

Setelah presentasi MVP dan diskusi awal dengan investor potensial, Product Owner/Manajemen PT Otomotif Digital Indonesia menyadari bahwa integrasi pembayaran digital akan menjadi nilai tambah yang signifikan untuk meningkatkan user experience dan model bisnis jangka panjang. Mereka mengajukan permintaan perubahan untuk menambahkan fitur pembayaran digital pada aplikasi Bengkelin.

Berikut adalah proses manajemen perubahan yang akan saya lakukan sebagai analis sistem:

1. Pencatatan Perubahan (Change Request Logging)

Setiap permintaan perubahan harus didokumentasikan secara formal.

- Aksi:
    - Menerima permintaan perubahan dari Product Owner secara resmi (misalnya, melalui email atau platform manajemen proyek seperti Jira, Trello, atau sejenisnya).
    - Mencatat detail permintaan dalam log perubahan kebutuhan (Change Request Log), termasuk:
        - ID Perubahan: CR-001 (Contoh: Change Request 001)
        - Judul Perubahan: Penambahan Fitur Pembayaran Digital
        - Diajukan Oleh: Product Owner / Manajemen PT Otomotif Digital Indonesia
        - Tanggal Pengajuan: 14 Juni 2025
        - Deskripsi Singkat: Integrasi metode pembayaran digital (misalnya: OVO, GoPay, transfer bank virtual account) untuk transaksi pemesanan servis kendaraan melalui aplikasi Bengkelin.
        - Alasan Pengajuan: Meningkatkan kenyamanan pengguna, mengoptimalkan proses pembayaran, dan mendukung monetisasi aplikasi di masa depan.
        - Prioritas Awal: Tinggi (disesuaikan setelah analisis dampak).

2. Analisis Dampak (Impact Analysis)

Ini adalah langkah krusial untuk memahami konsekuensi dari perubahan yang diusulkan terhadap seluruh aspek proyek.

- Aksi: Mengadakan pertemuan analisis dampak yang melibatkan:
    - Analis Sistem (memimpin analisis)
    - Product Owner (untuk klarifikasi visi)
    - Perwakilan Tim Pengembangan (Backend & Frontend)
    - Perwakilan Tim QA (Quality Assurance)

- Aspek yang Dianalisis:
    - Dampak pada Kebutuhan Fungsional (FR):
        - Penambahan FR baru: "Pengguna dapat memilih metode pembayaran digital."
        - Penambahan FR baru: "Pengguna dapat menyelesaikan pembayaran melalui gateway yang terintegrasi."
        - Penambahan FR baru: "Sistem dapat mengkonfirmasi status pembayaran ke bengkel dan pengguna."
        - Penambahan FR baru: "Bengkel dapat melihat status pembayaran pada pesanan."
        - Potensi modifikasi FR-3.4 (Konfirmasi Pemesanan) dan FR-6.4 (Update Status Servis).
    - Dampak pada Kebutuhan Non-Fungsional (NFR):
        - Keamanan (NFR-4): Peningkatan fokus pada keamanan transaksi, enkripsi data finansial, kepatuhan PCI DSS (jika relevan). Memerlukan lapisan keamanan tambahan.
        - Keandalan (NFR-2): Ketergantungan pada payment gateway eksternal. Memerlukan penanganan error yang robust jika payment gateway mengalami downtime.
        - Performa (NFR-1): Potensi latensi tambahan karena integrasi API eksternal.
        - Skalabilitas (NFR-3): Payment gateway yang dipilih harus mendukung volume transaksi yang besar.
    - Dampak pada Desain Sistem (Arsitektur):
        - Memerlukan modul integrasi payment gateway.
        - Perubahan pada database untuk menyimpan data transaksi dan status pembayaran.
        - Perubahan pada antarmuka pengguna (UI) untuk menampilkan opsi pembayaran.
    - Dampak pada Jadwal & Sumber Daya (Proyek):
        - Waktu: Penambahan waktu pengembangan (estimasi: 2-4 minggu untuk MVP, tergantung kompleksitas integrasi dan jumlah gateway). Ini berarti target rilis MVP dalam 2 bulan mungkin tidak tercapai atau fitur lain harus dikorbankan.
        - Biaya: Potensi biaya integrasi dengan payment gateway (biaya API, biaya transaksi).

        - Sumber Daya: Membutuhkan keahlian khusus dalam integrasi pembayaran atau testing keamanan.

    - Dampak pada Pengujian:

        - Penambahan test case untuk alur pembayaran, skenario kegagalan pembayaran, pengembalian dana (jika relevan).

    - Dampak pada Stakeholder Lain:

        - Bengkel perlu memahami alur pembayaran digital dan bagaimana mereka menerima pembayaran.
        - Admin perlu mengelola laporan transak

Tentu, mari kita simulasikan satu perubahan kebutuhan dan jelaskan proses manajemen perubahan yang akan dilakukan.

E. Manajemen Perubahan Kebutuhan

Dalam pengembangan software dengan pendekatan Agile, perubahan adalah hal yang tak terhindarkan dan seringkali bahkan disambut baik sebagai bentuk adaptasi terhadap kebutuhan pasar yang berkembang. Kunci utamanya adalah bagaimana kita mengelola perubahan tersebut secara terstruktur dan transparan.

Mari kita simulasikan skenario perubahan:

Skenario Perubahan Kebutuhan: Penambahan Fitur Pembayaran Digital

Setelah presentasi MVP dan diskusi awal dengan investor potensial, Product Owner/Manajemen PT Otomotif Digital Indonesia menyadari bahwa integrasi pembayaran digital akan menjadi nilai tambah yang signifikan untuk meningkatkan user experience dan model bisnis jangka panjang. Mereka mengajukan permintaan perubahan untuk menambahkan fitur pembayaran digital pada aplikasi Bengkelin.

Berikut adalah proses manajemen perubahan yang akan saya lakukan sebagai analis sistem:

---

1. Pencatatan Perubahan (Change Request Logging)

Setiap permintaan perubahan harus didokumentasikan secara formal.

- Aksi:
    - Menerima permintaan perubahan dari Product Owner secara resmi (misalnya, melalui email atau platform manajemen proyek seperti Jira, Trello, atau sejenisnya).

    - Mencatat detail permintaan dalam log perubahan kebutuhan (Change Request Log), termasuk:
        - ID Perubahan: CR-001 (Contoh: Change Request 001)
        - Judul Perubahan: Penambahan Fitur Pembayaran Digital
        - Diajukan Oleh: Product Owner / Manajemen PT Otomotif Digital Indonesia
        - Tanggal Pengajuan: 14 Juni 2025
        - Deskripsi Singkat: Integrasi metode pembayaran digital (misalnya: OVO, GoPay, transfer bank virtual account) untuk transaksi pemesanan servis kendaraan melalui aplikasi Bengkelin.
        - Alasan Pengajuan: Meningkatkan kenyamanan pengguna, mengoptimalkan proses pembayaran, dan mendukung monetisasi aplikasi di masa depan.
        - Prioritas Awal: Tinggi (disesuaikan setelah analisis dampak).

---

2. Analisis Dampak (Impact Analysis)

Ini adalah langkah krusial untuk memahami konsekuensi dari perubahan yang diusulkan terhadap seluruh aspek proyek.

- Aksi: Mengadakan pertemuan analisis dampak yang melibatkan:
    - Analis Sistem (memimpin analisis)
    - Product Owner (untuk klarifikasi visi)
    - Perwakilan Tim Pengembangan (Backend & Frontend)
    - Perwakilan Tim QA (Quality Assurance)
- Aspek yang Dianalisis:
    - Dampak pada Kebutuhan Fungsional (FR):
        - Penambahan FR baru: "Pengguna dapat memilih metode pembayaran digital."
        - Penambahan FR baru: "Pengguna dapat menyelesaikan pembayaran melalui gateway yang terintegrasi."
        - Penambahan FR baru: "Sistem dapat mengkonfirmasi status pembayaran ke bengkel dan pengguna."
        - Penambahan FR baru: "Bengkel dapat melihat status pembayaran pada pesanan."
        - Potensi modifikasi FR-3.4 (Konfirmasi Pemesanan) dan FR-6.4 (Update Status Servis).
    - Dampak pada Kebutuhan Non-Fungsional (NFR):

        - Keamanan (NFR-4): Peningkatan fokus pada keamanan transaksi, enkripsi data finansial, kepatuhan PCI DSS (jika relevan). Memerlukan lapisan keamanan tambahan.
        - Keandalan (NFR-2): Ketergantungan pada payment gateway eksternal. Memerlukan penanganan error yang robust jika payment gateway mengalami downtime.
        - Performa (NFR-1): Potensi latensi tambahan karena integrasi API eksternal.
        - Skalabilitas (NFR-3): Payment gateway yang dipilih harus mendukung volume transaksi yang besar.
    - Dampak pada Desain Sistem (Arsitektur):
        - Memerlukan modul integrasi payment gateway.
        - Perubahan pada database untuk menyimpan data transaksi dan status pembayaran.
        - Perubahan pada antarmuka pengguna (UI) untuk menampilkan opsi pembayaran.
    - Dampak pada Jadwal & Sumber Daya (Proyek):
        - Waktu: Penambahan waktu pengembangan (estimasi: 2-4 minggu untuk MVP, tergantung kompleksitas integrasi dan jumlah gateway). Ini berarti target rilis MVP dalam 2 bulan mungkin tidak tercapai atau fitur lain harus dikorbankan.
        - Biaya: Potensi biaya integrasi dengan payment gateway (biaya API, biaya transaksi).
        - Sumber Daya: Membutuhkan keahlian khusus dalam integrasi pembayaran atau testing keamanan.
    - Dampak pada Pengujian:
        - Penambahan test case untuk alur pembayaran, skenario kegagalan pembayaran, pengembalian dana (jika relevan).
    - Dampak pada Stakeholder Lain:
        - Bengkel perlu memahami alur pembayaran digital dan bagaimana mereka menerima pembayaran.
        - Admin perlu mengelola laporan transaksi.

---

3. Rekomendasi dan Presentasi ke Change Control Board (CCB) / Product Owner

Berdasarkan analisis dampak, analis sistem menyusun rekomendasi dan mempersiapkannya untuk diajukan ke otoritas yang berwenang (dalam konteks Agile, ini bisa jadi Product Owner dengan konsultasi manajemen senior).

- Aksi:
    - Mempresentasikan hasil analisis dampak kepada Product Owner (dan manajemen jika perubahan signifikan).
    - Rekomendasi:
        - Opsi 1 (Tidak Diakomodasi di MVP): Menunda fitur pembayaran digital ke fase pengembangan berikutnya (setelah MVP rilis), untuk menjaga target 2 bulan MVP. Keuntungan: Rilis MVP tepat waktu. Kerugian: Tidak mendapatkan manfaat pembayaran digital di awal.
        - Opsi 2 (Diakomodasi, dengan penyesuaian): Mengintegrasikan hanya satu payment gateway dasar di MVP, dan menggeser jadwal rilis MVP selama 2-4 minggu, atau menunda pengembangan 1-2 fitur lain yang kurang krusial dari MVP. Keuntungan: Mendapatkan manfaat pembayaran digital di awal. Kerugian: Penundaan rilis atau pengurangan fitur.
        - Opsi 3 (Diakomodasi, dengan tim tambahan): Jika sumber daya dan anggaran memungkinkan, merekrut atau menugaskan tim/sumber daya tambahan untuk mempercepat integrasi tanpa menunda rilis atau mengorbankan fitur lain.

---

4. Persetujuan Perubahan (Change Approval)

Keputusan akhir berada pada Product Owner atau Change Control Board (CCB) jika ada.

- Aksi:
    - Product Owner (setelah berdiskusi dengan manajemen) membuat keputusan berdasarkan rekomendasi dan prioritas bisnis.
    - Misalkan, Product Owner menyetujui Opsi 2: Mengintegrasikan satu payment gateway dasar, dan menerima penundaan rilis MVP selama 3 minggu.
    - Persetujuan dicatat secara formal dalam log perubahan (tanggal persetujuan, pihak yang menyetujui, keputusan akhir).

---

5. Implementasi Perubahan (Change Implementation)

Setelah disetujui, perubahan tersebut diintegrasikan ke dalam rencana pengembangan.

- Aksi:
    - Analis Sistem: Memperbarui dokumen SRS (terutama bagian Kebutuhan Fungsional dan Non-Fungsional) dan user stories yang relevan. Misalnya, menambahkan user stories seperti "Sebagai pengguna, saya ingin dapat membayar servis dengan OVO, sehingga saya tidak perlu transfer manual."
    - Product Owner: Memperbarui product backlog dan mengkomunikasikan perubahan prioritas dan jadwal kepada tim.
    - Tim Pengembangan: Memulai implementasi fitur pembayaran digital sesuai dengan spesifikasi yang diperbarui.
    - Tim QA: Merancang test case baru untuk pengujian fitur pembayaran.
    - Seluruh Tim: Terus berkomunikasi dan beradaptasi dengan perubahan yang disetujui.

Bagian F :

F. Penerapan Agile dalam Rekayasa Kebutuhan

Metode Agile, khususnya Scrum, sangat cocok untuk pengembangan aplikasi Bengkelin yang ingin merilis MVP dalam 2 bulan dan membutuhkan fleksibilitas terhadap perubahan. Dalam Agile, rekayasa kebutuhan bukanlah aktivitas satu kali di awal, melainkan proses berkelanjutan, iteratif, dan kolaboratif sepanjang siklus hidup proyek.

Berikut adalah bagaimana Agile diterapkan dalam siklus hidup kebutuhan Bengkelin:

1. Pembentukan Product Backlog

Product Backlog adalah daftar terurut semua fitur, fungsionalitas, perbaikan, dan pekerjaan lain yang diperlukan untuk produk. Ini adalah "sumber tunggal kebenaran" untuk semua pekerjaan yang akan dilakukan pada aplikasi Bengkelin. Item-item dalam backlog ini disebut Product Backlog Item (PBI), yang seringkali ditulis dalam format user story.

- Siapa yang Bertanggung Jawab: Product Owner bertanggung jawab penuh atas Product Backlog, termasuk isinya, urutannya, dan ketersediaannya bagi tim.

Namun, pengisian dan pemurnian (refinement) backlog adalah upaya kolaboratif.

- Proses:
    1. Pada awalnya, Product Owner akan bekerja dengan stakeholder (manajemen, calon pengguna, bengkel) untuk mengidentifikasi semua user stories berdasarkan analisis kebutuhan awal.
    2. Item-item akan diurutkan berdasarkan prioritas bisnis, nilai bagi pengguna, dan risiko.
    3. Backlog akan terus diperbarui dan diperhalus seiring berjalannya waktu melalui sesi backlog refinement.

2. Perencanaan Sprint (Sprint Planning) & Sprint Backlog

Setiap Sprint (periode waktu singkat dan tetap, misalnya 2 minggu) dimulai dengan Sprint Planning Meeting.

- Proses:
    - Product Owner mempresentasikan PBI prioritas tertinggi dari Product Backlog.
    - Development Team memperkirakan usaha yang diperlukan untuk PBI tersebut.
    - Bersama-sama, tim memilih sejumlah PBI yang realistis untuk diselesaikan dalam Sprint tersebut. PBI yang dipilih ini membentuk Sprint Backlog.
    - Tim akan memecah PBI menjadi tugas-tugas yang lebih kecil dan terkelola (tasks).

3. Prioritas Fitur (di Level MVP)

Dalam konteks MVP 2 bulan, prioritas fitur sangat ketat. Fokusnya adalah pada fungsi inti yang memberikan nilai terbesar dengan usaha minimal.

- Prioritas Tinggi (Harus Ada di MVP):
    - Registrasi & Login Pengguna
    - Pencarian Bengkel Berdasarkan Lokasi (termasuk melihat daftar dan detail singkat)
    - Melihat Detail Profil Bengkel (lengkap)
    - Pemesanan Layanan (memilih layanan, jadwal, konfirmasi)
    - Penerimaan/Penolakan Pesanan oleh Bengkel
    - Notifikasi Pesanan Baru (untuk bengkel)
    - Verifikasi Bengkel oleh Admin
- Prioritas Sedang (Diinginkan di MVP, Jika Waktu Memungkinkan):
    - Riwayat Servis Pengguna

    - Rating & Review Pengguna
    - Manajemen Profil Bengkel (update info, layanan)
    - Update Status Servis oleh Bengkel

- Prioritas Rendah (Setelah MVP):

    - Pembayaran Digital
    - Fitur Chat
    - Fitur Promo
    - Navigasi GPS

Product Owner akan memastikan Sprint pertama dan mungkin Sprint kedua berfokus pada PBI dengan prioritas tinggi untuk memastikan MVP dapat dirilis sesuai target.

4. Pertemuan Harian (Daily Standup / Daily Scrum)

Daily Standup adalah ritual harian singkat (maksimal 15 menit) di mana tim pengembang menyinkronkan aktivitas dan membuat rencana untuk 24 jam berikutnya. Ini membantu mengidentifikasi blocker dan menjaga fokus pada tujuan Sprint.

- Siapa yang Hadir: Seluruh Development Team, Scrum Master (fasilitator), dan Product Owner (opsional, sebagai pendengar).
- Waktu & Lokasi: Setiap hari di waktu dan tempat yang sama (misalnya, pukul 09:00 WIB di ruang rapat tim atau online call).
- Tiga Pertanyaan Kunci: Setiap anggota tim menjawab 3 pertanyaan berikut:

    1. Apa yang saya lakukan kemarin untuk membantu tim mencapai tujuan Sprint?
    2. Apa yang akan saya lakukan hari ini untuk membantu tim mencapai tujuan Sprint?
    3. Adakah hambatan (impediments) yang menghalangi saya atau tim dalam mencapai tujuan Sprint?

Contoh Product Backlog, Sprint Backlog, dan Prioritas Fitur

Product Backlog

Product Backlog adalah daftar dinamis, terurut, dan terus berkembang dari semua yang dibutuhkan dalam produk Bengkelin. Ini dikelola oleh Product Owner dan mencerminkan visi produk secara keseluruhan. Setiap item di sini disebut Product Backlog Item (PBI), yang biasanya ditulis sebagai user story. Prioritas diurutkan dari atas (tertinggi) ke bawah (terendah).

ID PBI	User Story	Prioritas	Estimasi (Story F	Keterangan
PBI-001	Sebagai penggu	Tinggi	3	Esensial untuk memulai.
PBI-002	Sebagai penggu	Tinggi	5	Fitur inti untuk penemuan bengkel.
PBI-003	Sebagai penggu	Tinggi	8	Penting untuk kepercayaan dan informasi.
PBI-004	Sebagai penggu	Tinggi	5	Bagian dari alur pemesanan inti.
PBI-005	Sebagai penggu	Tinggi	5	Bagian dari alur pemesanan inti.
PBI-006	Sebagai penggu	Tinggi	3	Finalisasi proses pemesanan.
PBI-009	Sebagai pemilik	Tinggi	3	Kritis untuk operasional bengkel.
PBI-010	Sebagai pemilik	Tinggi	5	Kritis untuk operasional bengkel.
PBI-012	Sebagai admin s	Tinggi	8	Kritis untuk menjaga kualitas platform.
PBI-007	Sebagai penggu	Sedang	8	Memberikan nilai jangka panjang. Mungkin di MVP, tergantung kapasitas.
PBI-008	Sebagai penggu	Sedang	5	Penting untuk kredibilitas bengkel. Mungkin di MVP.
PBI-011	Sebagai pemilik	Sedang	5	Memberikan transparansi kepada pengguna.
PBI-013	Sebagai penggu	Rendah	13	Contoh fitur yang mungkin ditunda ke fase selanjutnya atau MVP tahap 2.
...	(Fitur-fitur lainnya)			

Prioritas Fitur (untuk MVP dalam 2 Bulan)

Berdasarkan Product Backlog di atas, prioritas fitur untuk MVP Bengkelin akan sangat fokus pada fungsi inti:

- Prioritas Tinggi (Wajib ada di MVP): Fitur yang esensial agar aplikasi bisa berfungsi dan memberikan nilai dasar. Ini mencakup PBI-001, PBI-002, PBI-003, PBI-004, PBI-005, PBI-006, PBI-009, PBI-010, PBI-012.
    - Registrasi & Login Pengguna
    - Pencarian Bengkel & Melihat Detail Bengkel
    - Pemesanan Layanan (memilih layanan, jadwal, konfirmasi)
    - Notifikasi & Pengelolaan Pesanan (Terima/Tolak) untuk Bengkel
    - Verifikasi Bengkel oleh Admin
- Prioritas Sedang (Diinginkan di MVP, Jika Waktu Memungkinkan): Fitur yang menambah nilai signifikan tapi bukan penghalang utama untuk rilis. Ini bisa jadi PBI-007, PBI-008, PBI-011.
    - Riwayat Servis Pengguna
    - Rating & Review Pengguna
    - Update Status Servis oleh Bengkel
- Prioritas Rendah (Setelah MVP): Fitur yang akan dikembangkan setelah MVP dirilis, seperti PBI-013 (Pembayaran Digital) dan fitur lainnya.

Sprint Backlog

Sprint Backlog adalah subset dari Product Backlog yang dipilih oleh tim pengembang untuk diselesaikan dalam satu Sprint (misalnya, 2 minggu). Item-item di sini dipecah menjadi tugas-tugas (tasks) yang lebih kecil dan bisa dikerjakan secara individual.

Contoh Sprint Backlog (untuk Sprint 1, durasi 2 minggu):

Tujuan Sprint: Pengguna dapat mendaftar dan mencari bengkel terdekat, serta bengkel dapat menerima notifikasi pesanan.

ID PBI	User Story	Estimasi (Story F	Tugas (Tasks)	Status	Ditugaskan ke
PBI-001	Sebagai penggu	3	- Desain UI/UX h	In Progress	Bimo, Sarah, Arif, Rina
PBI-002	Sebagai penggu	5	- Desain UI/UX h	To Do	Sarah, Arif, Rina
PBI-009	Sebagai pemilik	3	- Desain mekani	To Do	Bimo, Sarah, Arif, Rina

Ilustrasi Sederhana Daily Standup

Daily Standup atau Daily Scrum adalah pertemuan singkat (15 menit) yang dilakukan setiap hari oleh tim pengembang untuk menyinkronkan aktivitas dan merencanakan pekerjaan untuk 24 jam ke depan.

Setting: Ruang rapat tim atau panggilan video, setiap hari pukul 09:00 WIB. Peserta:

- Bimo (Developer Backend)
- Sarah (Developer Frontend)
- Arif (Developer Backend/Database)
- Rina (QA Engineer)
- Scrum Master (fasilitator)
- Product Owner hadir sebagai pendengar

(Hari ke-3 Sprint 1, 14 Juni 2025, 09:00 WIB)

Scrum Master: "Selamat pagi tim! Mari kita mulai daily standup kita. Ingat, maksimal 15 menit, fokus pada tiga pertanyaan. Siapa yang mau mulai?"

Bimo (Dev Backend): "Pagi. Kemarin saya selesai mengimplementasikan endpoint API untuk registrasi pengguna (PBI-001) dan sudah melakukan unit testing di sisi saya. Hari ini, saya akan mulai mengerjakan logika backend untuk pemicu notifikasi pesanan baru (PBI-009). Sejauh ini tidak ada blocker."

Sarah (Dev Frontend): "Pagi. Kemarin saya berhasil mengintegrasikan UI halaman registrasi dengan API registrasi Bimo (PBI-001). Saat ini sudah bisa dites. Hari ini, saya akan mulai mendesain UI dan mengimplementasikan halaman pencarian bengkel (PBI-002). Tidak ada blocker."

Arif (Dev Backend/Database): "Pagi. Kemarin saya menyelesaikan desain skema database untuk data bengkel dan user (terkait PBI-001 & PBI-002). Hari ini, saya akan mulai mengerjakan logic untuk pengambilan data bengkel terdekat berdasarkan lokasi (PBI-002). Saya ada blocker kecil: Saya butuh klarifikasi dari Product Owner tentang apakah kita akan menggunakan GeoJSON atau hanya koordinat latitude/longitude biasa untuk data lokasi bengkel di database."

Rina (QA Engineer): "Pagi. Kemarin saya selesai membuat test case untuk alur registrasi dan login (PBI-001). Hari ini, saya akan mulai melakukan pengujian end-to-end untuk fitur registrasi. Blocker saya adalah saya belum mendapatkan akses ke lingkungan staging untuk pengujian ini."

Scrum Master: "Baik, terima kasih tim. Bimo dan Sarah, bagus progresnya. Arif, saya akan segera koordinasikan dengan Product Owner setelah standup ini untuk klarifikasi format data lokasi. Rina, saya akan pastikan tim DevOps memberikan akses staging untukmu segera setelah standup. Ada lagi yang ingin disampaikan atau blocker lain?"

(Tidak ada)

Scrum Master: “Oke, jika tidak ada lagi, selamat bekerja tim!”