Tugas Besar Milestone 3
Desain Taktis
II3160 - Integrated Systems Technology

Diampu oleh:
Daniel Wiyogo Dwiputro, S.T., M.T.

Disusun oleh :
Nakeisha Valya Shakila
18223133

PROGRAM STUDI SISTEM DAN TEKNOLOGI INFORMASI
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
JATINANGOR
2025

DAFTAR ISI

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Tahap ini merupakan kelanjutan dari proses perancangan sistem berbasis *Domain-Driven Design (DDD)* yang sebelumnya telah mencakup identifikasi domain, pembuatan subdomain, capability mapping, serta penyusunan bounded context dan context map. Pada tahap ini, kegiatan difokuskan pada **pembuatan glosarium Ubiquitous Language** dan **perancangan model domain** yang terdiri dari *Entities*, *Value Objects*, dan *Aggregates* menggunakan **Class Diagram**. Langkah ini bertujuan untuk menyatukan pemahaman konsep antara pihak pengembang dan pemangku kepentingan melalui penggunaan istilah yang konsisten, sekaligus merepresentasikan struktur logis domain dalam bentuk model konseptual yang jelas. Dengan demikian, sistem yang dirancang dapat memiliki bahasa bersama yang mudah dipahami serta fondasi model yang kuat, konsisten, dan mudah dikembangkan di tahap implementasi berikutnya.

1.2. Rumusan Masalah

Dari latar belakang yang ada, terdapat beberapa rumusan masalah yang diharapkan dapat terjawab setelah membaca laporan ini, antara lain sebagai berikut:

- Bagaimana membangun **Ubiquitous Language** yang dapat menjadi bahasa bersama antara pengembang dan pengguna agar tidak terjadi perbedaan pemahaman terhadap konsep domain?
- Bagaimana merancang **model domain** yang terdiri dari *Entities*, *Value Objects*, dan *Aggregates* untuk merepresentasikan logika bisnis sistem secara akurat dan terstruktur?
- Bagaimana memvisualisasikan hubungan antar komponen model domain tersebut melalui **Class Diagram** agar mudah dianalisis dan dikembangkan lebih lanjut?

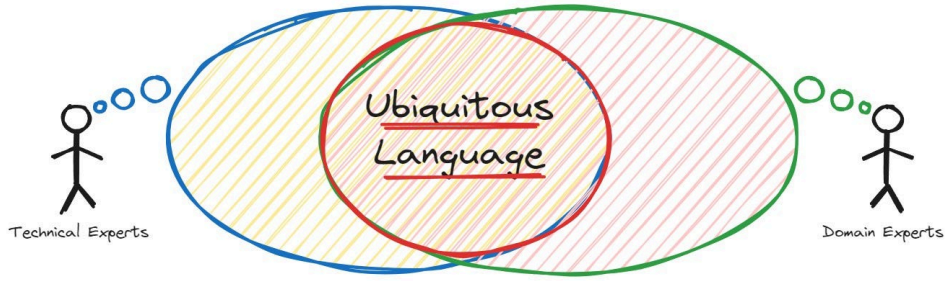
1.3. Tujuan

- Menyusun **glosarium Ubiquitous Language** yang berisi istilah-istilah penting dalam domain sistem beserta definisinya untuk memastikan keseragaman pemahaman.
- Merancang **model domain** yang terdiri dari *Entities*, *Value Objects*, dan *Aggregates* sebagai dasar logika bisnis sistem.
- Membuat **Class Diagram** untuk menggambarkan struktur hubungan antar elemen model domain secara visual, terorganisasi, dan mudah dipahami.

BAB II

PEMBAHASAN

2.1. Ubiquitous Language



Gambar 2.1 Ubiquitous Language

Pada tahap perancangan berbasis **Domain-Driven Design (DDD)**, **Ubiquitous Language** menjadi komponen penting yang menjembatani komunikasi antara pihak bisnis dan teknis. Bahasa ini menggambarkan istilah-istilah yang digunakan dalam domain sistem beserta maknanya, sehingga seluruh proses perancangan dan implementasi dapat menggunakan penamaan yang konsisten. Dalam proses ini, **domain expert** berperan sebagai pihak yang memahami secara mendalam konteks bisnis, proses, serta kebutuhan nyata dari sistem yang akan dibangun, sedangkan **technical expert** berperan untuk menerjemahkan konsep-konsep tersebut ke dalam bentuk desain arsitektur dan implementasi sitem. Kolaborasi antara keduanya memastikan bahwa solusi yang dikembangkan tidak hanya tepat secara teknis, tetapi juga sesuai dengan kebutuhan dan logika bisnis yang ada. Pada bagian ini, istilah seperti Trip, Booking, Participant, dan Transaction merupakan istilah inti dalam Booking Context, sedangkan istilah lain seperti Financial Report, Notification, Feedback, Rating, dan Support Ticket berasal dari konteks berbeda namun tetap dicantumkan karena berinteraksi dengan proses pemesanan. Berikut merupakan glosarium dari istilah yang ada pada perancangan sistem yang saya buat :

Tabel 1 Glosarium Ubiquitous Language

Istilah	Deskripsi Singkat
Trip	Paket perjalanan wisata yang mencakup jadwal, itinerary, kapasitas, dan pemandu.
Schedule	Jadwal perjalanan yang mencatat waktu mulai, waktu selesai, dan lokasi kegiatan.
Itinerary	Rangkaian tujuan dan aktivitas yang dilakukan selama perjalanan.
Guide	Pemandu wisata yang bertanggung jawab dalam mendampingi peserta selama trip.
Booking	Pemesanan trip oleh peserta yang mencakup status, detail peserta, dan transaksi pembayaran.
Participant	Peserta yang mengikuti trip dan memiliki data pribadi serta riwayat pemesanan.
Transaction	Catatan transaksi pembayaran yang dilakukan peserta untuk trip tertentu.

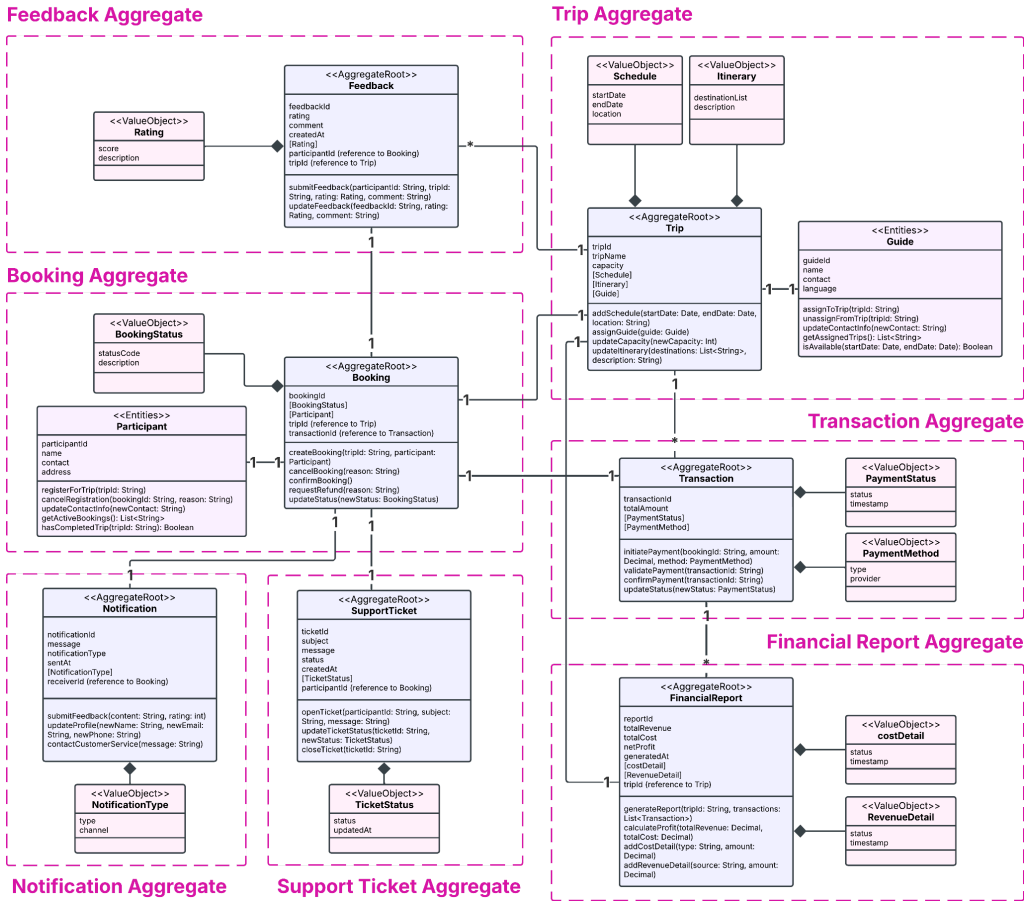
PaymentStatus	Nilai yang menunjukkan kondisi terkini dari transaksi (misalnya: Pending, Sukses, Gagal).
PaymentMethod	Jenis metode pembayaran yang digunakan (contoh: Transfer Bank, E-Wallet, Kartu Kredit).
Financial Report	Laporan keuangan yang berisi ringkasan pendapatan, biaya, dan laba bersih dari pelaksanaan trip.
Revenue Detail	Rincian sumber pendapatan yang diperoleh dari pelaksanaan trip.
Cost Detail	Rincian biaya operasional yang dikeluarkan selama penyelenggaraan trip.
Feedback	Umpan balik atau ulasan dari peserta mengenai pengalaman mengikuti trip.
Rating	Skor penilaian kuantitatif yang diberikan peserta terhadap kualitas trip.
Notification	Sistem pesan yang digunakan untuk mengirimkan informasi kepada peserta atau admin.
Notification Type	Jenis notifikasi berdasarkan saluran komunikasi (misalnya: Email, WhatsApp, In-App).
Support Ticket	Tiket bantuan yang dibuat oleh peserta untuk meminta dukungan atau melaporkan kendala.
Ticket Status	Status tiket bantuan (misalnya: Open, In Progress, Closed).
Booking Status	Kondisi atau tahap pemesanan (misalnya: Confirmed, Cancelled, Refunded).

2.2. Merancang Model

Merancang model menggunakan **Class Diagram** merupakan langkah penting dalam menggambarkan struktur dan hubungan antar komponen di dalam domain sistem. Melalui *Class Diagram*, keterkaitan antar kelas, atribut, dan operasi dapat divisualisasikan secara sistematis sehingga mempermudah pemahaman terhadap interaksi dan alur logika sistem. Dalam konteks ini, pemodelan difokuskan pada elemen-elemen domain seperti *Entities*, *Aggregates*, dan *Value Objects*. Diagram ini disusun berdasarkan prinsip *Domain-Driven Design (DDD)*, yang menekankan pentingnya representasi eksplisit konsep bisnis dalam struktur model maupun kode sistem.

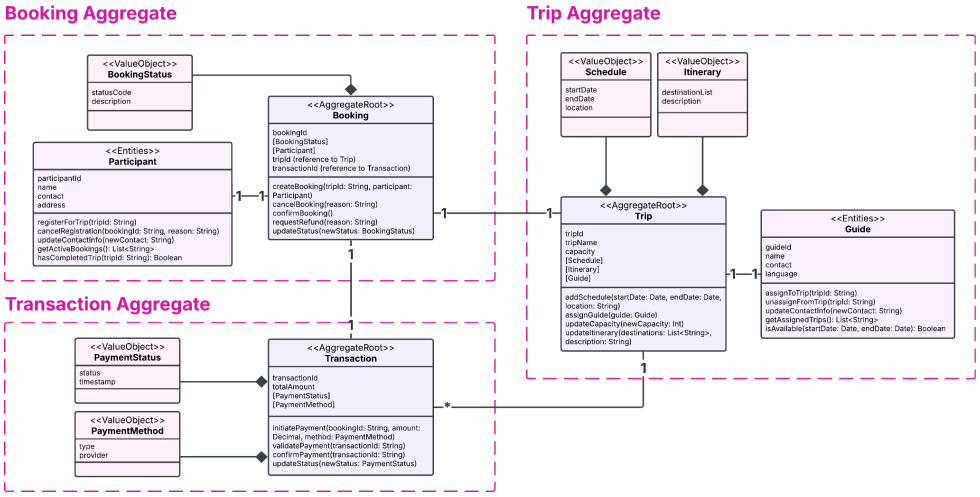
- Entity : merepresentasikan objek dengan identitas unik yang keberadaannya perlu dipertahankan,
- Value Object : menggambarkan nilai atau atribut yang tidak memiliki identitas tersendiri.
- Aggregate : pengelompokan dari *entity* dan *value object* yang saling berkaitan dan dikendalikan oleh satu Aggregate Root untuk menjaga konsistensi data serta perilaku domain.

Relasi antara **Aggregate Root** dan **Value Object** selalu bersifat **composition**, karena Value Object sepenuhnya melekat pada siklus hidup Aggregate Root dan tidak memiliki relasi seperti one-to-one atau one-to-many secara terpisah. Berikut merupakan class diagram domain yang menggambarkan struktur dan hubungan antar komponen berdasarkan prinsip *Domain-Driven Design (DDD)*.



Gambar 2.2 Class Diagram DDD Principle

Perlu diperhatikan bahwa aggregate seperti Feedback, Notification, SupportTicket, dan FinancialReport yang muncul pada Gambar 2.2 tidak termasuk dalam ruang lingkup implementasi pada Milestone 4. Oleh karena itu, class diagram berikut berfokus pada core context "**Booking Context**" tanpa melibatkan keempat aggregate tersebut.



Gambar 2.3 Core Context Class Diagram DDD Principle

Pada *core context* ini, model domain dibangun dengan berfokus pada **tiga aggregate** saja, yaitu **Booking**, **Trip**, dan **Transaction**. Ketiga aggregate tersebut dipilih karena secara langsung merepresentasikan proses utama dalam sistem.

Tabel 2 Daftar Atribut

Komponen	Atribut	Tipe Data	Deskripsi
Notification	notificationId	Int	Identitas unik setiap notifikasi.
	message	String	Isi pesan notifikasi.
	notificationType	String	Jenis notifikasi (mis. pengingat, konfirmasi).
	sentAt	DateTime	Waktu pengiriman notifikasi.
	receiverId	Int	Mengacu pada pemesan yang menerima notifikasi.
NotificationType	type	String	Jenis notifikasi (Email, SMS, In-App).
	channel	String	Media pengiriman notifikasi.
SupportTicket	ticketId	Int	Identitas unik tiket bantuan.
	subject	String	Topik atau judul masalah.
	message	Text	Isi pesan atau deskripsi permasalahan.
	status	String	Status tiket (Open, In Progress, Closed).
	createdAt	DateTime	Waktu tiket dibuat.
	participantId	Int	Mengacu pada peserta yang membuat tiket.
TicketStatus	status	String	Status tiket saat ini.
	updatedAt	DateTime	Waktu terakhir status diperbarui.
Participant	participantId	Int	Identitas unik peserta.
	name	String	Nama peserta.
	contact	String	Informasi kontak (telepon/email).
	address	String	Alamat tempat tinggal peserta.
Booking	bookingId	Int	Identitas unik pemesanan.
	tripId	Int	Referensi ke trip yang dipesan.

	transactionId	Int	Referensi ke transaksi pembayaran.
BookingStatus	statusCode	String	Kode status (PENDING, CONFIRMED, CANCELLED).
	description	String	Penjelasan status pemesanan.
Feedback	feedbackId	Int	Identitas unik feedback.
	rating	Int	Nilai penilaian (misal 1–5).
	comment	Text	Komentar dari pengguna.
	createdAt	DateTime	Waktu pembuatan feedback.
	participantId	Int	Referensi ke peserta pemberi feedback.
	tripId	Int	Referensi ke trip yang dinilai.
Rating	score	Int	Skor penilaian (1–5).
	description	String	Penjelasan makna skor.
Schedule	startDate	Date	Tanggal mulai perjalanan.
	endDate	Date	Tanggal selesai perjalanan.
	location	String	Lokasi kegiatan atau titik kumpul.
Itinerary	destinationList	Array<String>	Daftar destinasi yang dikunjungi.
	description	Text	Penjelasan susunan kegiatan perjalanan.
Trip	tripId	Int	Identitas unik trip.
	tripName	String	Nama atau judul paket trip.
	capacity	Int	Kapasitas maksimum peserta.
Guide	guideId	Int	Identitas unik pemandu.
	name	String	Nama pemandu wisata.
	contact	String	Kontak pemandu.
	language	String	Bahasa yang dikuasai.
Transaction	transactionId	Int	Identitas unik transaksi.
	totalAmount	Decimal	Total nominal pembayaran.
PaymentStatus	status	String	Status pembayaran (Success, Pending, Failed).

	timestamp	DateTime	Waktu pembaruan status.
PaymentMethod	type	String	Jenis metode (transfer, e-wallet, kartu kredit).
	provider	String	Penyedia layanan pembayaran (Bank, OVO, dll).
FinancialReport	reportId	Int	Identitas unik laporan keuangan.
	totalRevenue	Decimal	Total pendapatan.
	totalCost	Decimal	Total biaya.
	netProfit	Decimal	Laba bersih.
	generatedAt	DateTime	Waktu laporan dibuat.
	tripId	Int	Trip yang menjadi dasar laporan.
CostDetail	status	String	Jenis atau kategori biaya.
	timestamp	DateTime	Waktu pencatatan biaya.
RevenueDetail	status	String	Jenis atau sumber pendapatan.
	timestamp	DateTime	Waktu pencatatan pendapatan.

Tabel 2 Daftar Method

Komponen	Nama Method	Deskripsi Singkat
Trip	addSchedule(startDate: Date, endDate: Date, location: String) : Void	Menambahkan jadwal perjalanan baru ke dalam trip.
	assignGuide(guide: Guide) : Void	Menetapkan pemandu wisata (guide) ke trip tertentu.
	updateCapacity(newCapacity: Int) : Void	Memperbarui kapasitas maksimum peserta trip.
	updateItinerary(destinations: List, description: String) : Void	Memperbarui daftar tujuan dan deskripsi itinerary trip.
Guide	assignToTrip(tripId: String) : Void	Menugaskan guide ke sebuah trip berdasarkan ID trip.
	unassignFromTrip(tripId: String) : Void	Melepas penugasan guide dari sebuah trip.
	updateContactInfo(newContact: String) : Void	Memperbarui informasi kontak guide.

	getAssignedTrips() : List	Mengambil daftar trip yang sedang ditangani oleh guide.
	isAvailable(startDate: Date, endDate: Date) : Boolean	Mengecek ketersediaan guide pada rentang tanggal tertentu.
Feedback	submitFeedback(participantId: String, tripId: String, rating: Rating, comment: String) : Void	Mengirimkan ulasan dan penilaian terhadap trip yang diikuti.
	updateFeedback(feedbackId: String, rating: Rating, comment: String) : Void	Memperbarui isi atau nilai dari feedback yang telah diberikan.
Transaction	initiatePayment(bookingId: String, amount: Decimal, method: PaymentMethod) : Transaction	Memulai proses pembayaran untuk booking tertentu.
	validatePayment(transactionId: String) : Boolean	Memvalidasi status transaksi pembayaran.
	confirmPayment(transactionId: String) : Void	Mengonfirmasi bahwa pembayaran berhasil diproses.
	updateStatus(newStatus: PaymentStatus) : Void	Memperbarui status pembayaran dalam sistem.
FinancialReport	generateReport(tripId: String, transactions: List) : FinancialReport	Menghasilkan laporan keuangan untuk trip tertentu.
	calculateProfit(totalRevenue: Decimal, totalCost: Decimal) : Decimal	Menghitung laba bersih berdasarkan pendapatan dan biaya.
	addCostDetail(type: String, amount: Decimal) : Void	Menambahkan rincian biaya ke laporan keuangan.
	addRevenueDetail(source: String, amount: Decimal) : Void	Menambahkan rincian pendapatan ke laporan keuangan.
Booking	createBooking(tripId: String, participant: Participant) : Booking	Membuat pemesanan baru untuk sebuah trip oleh peserta.
	cancelBooking(reason: String) : Void	Membatalkan pemesanan dengan alasan tertentu.
	confirmBooking() : Void	Mengonfirmasi bahwa pemesanan telah disetujui.
	requestRefund(reason: String) : Void	Mengajukan permintaan pengembalian dana.
	updateStatus(newStatus: BookingStatus) : Void	Memperbarui status pemesanan berdasarkan perubahan kondisi.

Participant	registerForTrip(tripId: String) : Void	Mendaftarkan diri ke dalam trip tertentu.
	cancelRegistration(bookingId: String, reason: String) : Void	Membatalkan keikutsertaan dalam trip dengan alasan tertentu.
	updateContactInfo(newContact: String) : Void	Memperbarui informasi kontak peserta.
	getActiveBookings() : List	Mengambil daftar booking aktif yang dimiliki peserta.
	hasCompletedTrip(tripId: String) : Boolean	Mengecek apakah peserta telah menyelesaikan trip tertentu.
SupportTicket	openTicket(participantId: String, subject: String, message: String) : SupportTicket	Membuka tiket bantuan baru oleh peserta.
	updateTicketStatus(ticketId: String, newStatus: TicketStatus) : Void	Memperbarui status tiket bantuan.
	closeTicket(ticketId: String) : Void	Menutup tiket bantuan yang telah diselesaikan.
Notification	submitFeedback(content: String, rating: Int) : Void	Mengirimkan umpan balik pengguna melalui sistem notifikasi.
	updateProfile(newName: String, newEmail: String, newPhone: String) : Void	Memperbarui profil pengguna terkait notifikasi.
	contactCustomerService(message: String) : Void	Menghubungi layanan pelanggan melalui pesan notifikasi.

BAB III
PENUTUP

3.1. Kesimpulan

Tahap perancangan sistem berbasis **Domain-Driven Design (DDD)** ini berhasil menghasilkan **Ubiquitous Language** yang menjadi bahasa bersama antara pengembang dan pemangku kepentingan, serta **model domain** yang terstruktur melalui **Entities**, **Value Objects**, dan **Aggregates**. Pendekatan DDD memberikan pemahaman yang konsisten terhadap konsep bisnis dan memastikan setiap elemen sistem merepresentasikan logika domain secara jelas. Dari hasil perancangan dua class diagram, dapat disimpulkan bahwa **class diagram berbasis DDD** lebih sesuai digunakan karena menonjolkan pemodelan konsep bisnis secara mendalam dan mampu beradaptasi dengan perubahan kebutuhan sistem. Dengan demikian, rancangan ini menjadi fondasi kuat untuk pengembangan sistem yang konsisten, mudah dikembangkan, dan tetap selaras dengan tujuan bisnis.

DAFTAR PUSTAKA

Jamesmontemagno. (n.d.). Implementing value objects - .NET. Microsoft Learn. https://learn.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/implement-value-objects

Jammeli, M. Y. (2024, June 5). The power of ubiquitous language in software development: simplified. Medium. https://medium.com/@jammelimohamedyassin/the-power-of-ubiquitous-language-in-software-development-simplified-38c5148c02c4

Pccontrol. (2013, October 10). Pengetahuan Dasar dan contoh Diagram Kelas (class diagram). DASAR KOMPUTER BUAT PEMULA. http://pccontrol.wordpress.com/2013/01/21/pengetahuan-dasar-dan-contoh-diagram-kelas-class-diagram/

Examples. (2025, October 17). Context Mapper. Retrieved November 7, 2025, from https://contextmapper.org/docs/examples/

Alokmishra, V. a. P. B. (2025, May 24). DDD Context Mapping by example: Policy Management. Alok Mishra. Retrieved November 7, 2025, from https://alok-mishra.com/2021/06/29/ddd-context-mapping-by-example-policy-management/