AI-Driven E-Commerce (Master's Level)
Book Structure & 14-Week Plan (Placeholder)

Faculty of Computers and Information Sciences
Mansoura University, Egypt

February 11, 2026

Contents

How to Use This Draft

This document is a *structure-only* placeholder for an AI-driven, practical Master's-level e-commerce book. Each chapter will be written later; for now, sections contain brief bullets stating scope, key skills, and deliverables.

Intended Audience and Prerequisites

- Audience: Master's students in Information Systems / Computer & Information Sciences.
- Prereqs (recommended): databases, basic programming, web fundamentals, and introductory ML.
- Tooling (suggested): Python, notebooks, APIs, cloud services, and an ERP/EA modeling tool (as available).

Hands-on Track (Agreed)

- Approach: both tracks (lighter each).
- Track A (Enterprise/ERP): Odoo-focused integration labs (Sales/Inventory/Accounting flows).
- Track B (AI/ML): ML-heavy notebooks using e-commerce datasets (recsys, pricing, fraud, support).

Course Outcomes (Draft)

- Design end-to-end e-commerce systems with enterprise architecture (EA) and integration concerns.
- Apply ML/DL/AI to personalization, search, pricing, fraud, operations, and customer service.
- Integrate e-commerce with ERP/CRM/SCM (reference: Odoo) via APIs, events, and modern integration patterns.
- Evaluate systems for security, privacy, governance, ethics, and measurable business value.

Part I

14-Week Teaching Plan (Mapping Weeks to Chapters)

Chapter 1

Weekly Plan Overview

Course emphasis (agreed)

- Overall emphasis: balanced (EA/ERP + ML/DL/AI).
- ERP reference stack: Odoo (concepts remain vendor-neutral where possible).

Week	Topic (maps to the corresponding chapter)
1	Foundations of modern e-commerce systems and AI-driven product thinking
2	Digital platforms, marketplaces, and business models (B2C/B2B/D2C, multi-sided platforms)
3	E-commerce data foundations: events, tracking, data quality, and experimentation
4	Enterprise architecture for e-commerce: capabilities, domains, and reference architectures
5	Enterprise integration: APIs, iPaaS, ESB, events/streaming, and integration patterns
6	ERP/CRM/SCM integration: order-to-cash, procure-to-pay, inventory, and master data
7	Recommenders I: classical + learning-to-rank for personalization and search
8	Recommenders II: deep learning, embeddings, sequence models, and retrieval
9	Pricing and promotion analytics: forecasting, elasticity, and optimization (ML + OR)
10	Fraud, risk, and trust & safety: anomaly detection, graph ML, and AML patterns
11	Customer service automation: LLMs, RAG, chatbots, and agentic workflows
12	MLOps + DataOps for e-commerce: deployment, monitoring, drift, and governance
13	Security, privacy, compliance, and responsible AI in commerce
14	Capstone: enterprise-grade AI e-commerce solution architecture + evaluation

Part II

Book Structure (Chapters with Placeholders)

Chapter 2

An Introduction to E-Commerce with a Focus on Machine Learning, Deep Learning, and Artificial Intelligence

Abstract

Electronic commerce (e-commerce) has evolved from static online catalogs into highly dynamic, data-driven ecosystems. This transformation has been powered largely by advances in artificial intelligence (AI), particularly machine learning (ML) and deep learning (DL). This chapter introduces the foundations of e-commerce and explains how AI techniques are embedded across the e-commerce value chain—from customer acquisition and product discovery to pricing, logistics, and fraud prevention. The discussion is aimed at master's students in computer and information sciences, and therefore emphasizes conceptual clarity, system-level thinking, and the mapping between theoretical models and real-world e-commerce applications.

Keywords: e-commerce, artificial intelligence, machine learning, deep learning, recommendation systems, dynamic pricing, fraud detection, personalization, logistics optimization

2.1 Introduction to E-Commerce

E-commerce refers to the buying and selling of goods and services via electronic networks, primarily the internet [**?**]. It encompasses a broad set of transactional models: business-to-consumer (B2C) online retail, business-to-business (B2B) procurement platforms, consumer-to-consumer (C2C) marketplaces, and consumer-to-business (C2B) models such as influencer platforms and freelance marketplaces [**?**].

From a computing perspective, an e-commerce platform is not just a web front-end. It is a socio-technical system composed of:

- User interfaces (web, mobile, conversational)

- Application logic (catalog, cart, checkout, account management)
- Payment and risk engines
- Logistics and fulfillment systems
- Data infrastructure and analytics pipelines
- AI/ML components that drive personalization, prediction, and automation

A key characteristic of e-commerce is its data richness. Every user interaction—page views, searches, clicks, scrolls, wishlists, add-to-carts, purchases, returns, and even time-to-decision—can be captured as fine-grained behavioral logs. Combined with product metadata, prices, promotions, and external contextual data, this creates an ideal environment for AI and ML [**? ?**].

2.1.1 E-Commerce Business Models and Value Chain

Common e-commerce models include:

- **B2C retail:** Online stores selling directly to consumers.
- **Marketplaces:** Platforms mediating between many sellers and buyers.
- **B2B platforms:** Portals for wholesale ordering, procurement, and supply chain integration.
- **Digital services:** Software subscriptions, digital content, and online education.

Across these models, the **e-commerce value chain** typically includes:

1. Customer acquisition and traffic generation (marketing, ads, SEO)
2. Product discovery and decision support (search, recommendations, reviews)
3. Conversion and transaction processing (checkout, payments, risk checks)
4. Order fulfillment and logistics (inventory, warehousing, routing)
5. Post-purchase engagement (support, returns, loyalty, re-activation)

AI, ML, and DL are now present at each stage of this chain [**?**].

2.2 Foundations of AI, Machine Learning, and Deep Learning

2.2.1 Definitions and Relationships

- **Artificial Intelligence (AI):** Systems that exhibit behaviors considered “intelligent” (perception, reasoning, learning, decision-making).

- **Machine Learning (ML):** Algorithms that learn patterns from data and improve with experience.
- **Deep Learning (DL):** ML methods based on multi-layer neural networks that learn representations from large-scale data.

Thus, deep learning $\subset$ machine learning $\subset$ artificial intelligence [**? ? ?**].

2.2.2 Types of Learning and E-Commerce Examples

1. **Supervised learning:** learn $f : X \to Y$ from labeled examples (x_i, y_i). Examples: conversion prediction, fraud classification, demand forecasting.
2. **Unsupervised learning:** discover structure in unlabeled data. Examples: customer segmentation, anomaly detection, and learning embeddings from interactions.
3. **Reinforcement learning:** learn a policy to maximize long-term reward. Examples: recommendation policies and dynamic pricing under constraints.
4. **Representation learning:** learn features $\phi(x)$ automatically (often via DL). Examples: user/item embeddings; Transformer-based session representations.

2.3 Machine Learning Across the E-Commerce Lifecycle

2.3.1 Personalized Recommendations

Recommendation systems use past behavior and product attributes to suggest items a user is likely to buy [**? ?**]. Key evaluation ideas include ranking quality (precision@k, recall@k, NDCG) and business impact (incremental revenue) [**?**].

2.3.2 Search and Ranking

Learning-to-rank uses user feedback (including clicks) to optimize product ranking in response to queries [**?**]. Modern systems increasingly incorporate semantic retrieval and NLP components [**?**].

2.3.3 Segmentation, churn, and CLV

Segmentation (e.g., via clustering) and predictive modeling (e.g., churn/CLV) connect behavioral data to marketing and retention actions [**?**].

2.3.4 Dynamic Pricing and Promotion Optimization

Pricing combines prediction (demand response) with optimization under constraints; evaluation typically requires controlled online experimentation [**? ?**].

2.3.5 Demand Forecasting and Inventory Management

Forecasting methods range from statistical time-series models to ML sequence models; forecasts drive replenishment and inventory decisions [? ?].

2.3.6 Fraud Detection and Transaction Security

Fraud detection is an imbalanced classification problem with non-stationarity (attackers adapt), and must balance false positives against fraud loss [? ?].

2.3.7 Customer Service and Conversational Commerce

Customer support automation mixes information retrieval, intent/entity extraction, and increasingly LLM-based interaction [?].

2.4 Deep Learning in E-Commerce

DL is especially useful for unstructured and high-dimensional data (text, images, sequences) [?].

2.4.1 Neural Recommendation Systems

Deep recommender architectures model non-linear user–item interactions and session dynamics [? ?].

2.4.2 NLP for Product Text and Reviews

NLP supports query understanding, semantic search, review analysis, and conversational interfaces [?].

2.4.3 Computer Vision and Visual Search

Vision models support category classification, attribute extraction, and similarity search via embeddings [?].

2.5 Data, Architecture, and Deployment Considerations

E-commerce ML requires robust pipelines (data quality, governance), reliable deployment (latency, availability), and continuous evaluation (monitoring and experiments) [?].

2.6 Challenges and Responsible AI

Key challenges include bias, privacy/security, explainability for high-impact decisions, and the organizational processes needed for safe deployment [**? ?**].

2.7 Multiple-choice questions (MCQs)

1. Which relationship is correct?
   (a) AI $\subset$ ML $\subset$ DL
   (b) DL $\subset$ ML $\subset$ AI
   (c) ML $\subset$ DL $\subset$ AI
   (d) AI, ML, and DL are disjoint fields
2. In e-commerce, which is **most** naturally framed as a *ranking* problem?
   (a) Choosing the best warehouse location for a new facility
   (b) Ordering products in a search results page for a query
   (c) Estimating the total demand for next quarter
   (d) Balancing accounting entries for an invoice
3. Which metric is **most** aligned with offline evaluation of a top-k recommender?
   (a) NDCG@k
   (b) Mean squared error (MSE) on prices
   (c) Page load time (ms)
   (d) Uptime percentage
4. Fraud detection is often difficult because:
   (a) Fraud labels are always perfectly accurate
   (b) The problem is typically extremely imbalanced and attackers adapt over time
   (c) Fraud has no economic cost, only technical cost
   (d) The best approach is always rule-based
5. Which statement best describes why A/B testing is important in e-commerce ML?
   (a) It guarantees the model is unbiased
   (b) It measures causal impact on business KPIs under real user behavior
   (c) It replaces the need for offline evaluation entirely
   (d) It eliminates the need for monitoring after deployment

Answer key

1. (b)
2. (b)
3. (a)
4. (b)
5. (b)

Chapter 3

An Introduction to E-Commerce with a Focus on Machine Learning, Deep Learning, and Artificial Intelligence

Electronic commerce (e-commerce) has evolved from static online catalogs into highly dynamic, data-driven ecosystems. This transformation has been powered largely by advances in artificial intelligence (AI), particularly machine learning (ML) and deep learning (DL). This chapter introduces the foundations of e-commerce and explains how AI techniques are embedded across the e-commerce value chain—from customer acquisition and product discovery to pricing, logistics, and fraud prevention. The discussion is aimed at master's students in computer and information sciences, and therefore emphasizes conceptual clarity, system-level thinking, and the mapping between theoretical models and real-world e-commerce applications.

Keywords: E-commerce, Artificial Intelligence, Machine Learning, Deep Learning, Recommendation Systems, Dynamic Pricing, Fraud Detection, Personalization, Logistics Optimization

3.1 Introduction to E-Commerce

E-commerce refers to the buying and selling of goods and services via electronic networks, primarily the internet. It encompasses a broad set of transactional models: business-to-consumer (B2C) online retail, business-to-business (B2B) procurement platforms, consumer-to-consumer (C2C) marketplaces, and consumer-to-business (C2B) models such as influencer platforms and freelance marketplaces.

[?]

[?]

From a computing perspective, an e-commerce platform is not just a web front-end. It is a socio-technical system composed of:

- User interfaces (web, mobile, conversational)

- Application logic (catalog, cart, checkout, account management)
- Payment and risk engines
- Logistics and fulfillment systems
- Data infrastructure and analytics pipelines
- AI/ML components that drive personalization, prediction, and automation

A key characteristic of e-commerce is its data richness. Every user interaction—page views, searches, clicks, scrolls, wishlists, add-to-carts, purchases, returns, and even time-to-decision—can be captured as fine-grained behavioral logs.Combined with product metadata, prices, promotions, and external contextual data, this creates an ideal environment for AI and ML. [**?**]

3.1.1 E-Commerce Business Models and Value Chain

Common e-commerce models include:

- **B2C Retail:** Online stores selling directly to consumers (e.g., Amazon, local grocery delivery).
- **Marketplaces:** Platforms mediating between many sellers and buyers (e.g., eBay, regional platforms).
- **B2B Platforms:** Portals for wholesale ordering, procurement, and supply chain integration.
- **Digital Services:** Software subscriptions, digital content, and online education.

Across these models, the **e-commerce value chain** typically includes:

1. Customer acquisition and traffic generation (marketing, ads, SEO).
2. Product discovery and decision support (search, recommendations, reviews).
3. Conversion and transaction processing (checkout, payments, risk checks).
4. Order fulfillment and logistics (inventory, warehousing, routing).
5. Post-purchase engagement (support, returns, loyalty, re-activation

AI, ML, and DL are now present at each stage of this chain.[**?**]

3.2 Foundations of AI, Machine Learning, and Deep Learning

3.2.1 Definitions and Relationships

- **Artificial Intelligence (AI):** The broader field focused on building systems that exhibit behaviors considered “intelligent,” such as perception, reasoning, learning, and decision-making.

- **Machine Learning (ML):** A subfield of AI that develops algorithms which learn patterns from data and improve with experience, rather than relying on explicit, hand-crafted rules.
- **Deep Learning (DL):** A subfield of ML based on artificial neural networks with many layers, capable of automatically learning complex representations from large volumes of data, particularly unstructured data such as images, text, and click sequences.

ThThus, deep learning $\subset$ machine learning $\subset$ artificial intelligence. [**? ? ?**]

3.2.2 Types of Learning and E-Commerce Examples

1. **Supervised learning:** Learn a mapping $f : X \to Y$ from labeled examples (x_i, y_i). Examples: conversion prediction, fraud classification, demand forecasting.
2. **Unsupervised learning:** Discover structure in unlabeled data. Examples: customer clustering, anomaly detection, learning product embeddings from co-view/co-purchase graphs.
3. **Reinforcement learning (RL):** Learn a policy that maps states to actions to maximize long-term reward. Examples: adaptive recommendation strategies; dynamic pricing agents balancing revenue, churn, and fairness constraints.
4. **Representation learning (often via DL):** Learn feature representations $\phi(x)$ automatically. Examples: user/item embeddings; Transformer-based session models.

AI, ML, and DL are now present at each stage of this chain.[**?**]

Chapter 4

Platforms, Marketplaces, and Business Models (Week 2)

4.1 Learning objectives

- Distinguish e-commerce *channels* (storefronts) from *platforms* (ecosystems) and *marketplaces* (multi-seller).
- Analyze platform economics: network effects, multi-homing, and pricing of participation.
- Translate business model choices into architectural requirements (identity, trust, data, integration, ERP).

4.2 From storefront to platform

- **Storefront:** a single merchant selling to buyers (B2C/B2B) via a web/app channel.
- **Platform:** a system that enables interactions among multiple participant groups (e.g., buyers, sellers, advertisers, logistics providers).
- **Marketplace:** a platform where multiple sellers list items and transactions are mediated by platform policies and infrastructure.

4.3 Value creation, value capture, and growth

4.3.1 Value creation and matching

Platforms create value by reducing search and transaction costs, increasing variety, and improving trust; AI frequently strengthens these effects by improving matching and decision support.

4.3.2 Value capture models

- Take rate (commission on completed transactions)
- Subscription (seller or buyer membership tiers)
- Listing fees and value-added services (e.g., promotions, analytics)
- Advertising (sponsored listings, retail media)
- Fulfillment fees (warehousing, last-mile delivery, returns handling)

4.3.3 Network effects

- **Direct network effects:** more users attract more users (e.g., social commerce communities).
- **Indirect network effects:** more buyers attract more sellers and vice versa (classic marketplace dynamic).
- **Cold start:** bootstrapping supply and demand when network effects are weak.
- **Multi-homing:** sellers/buyers participate in multiple platforms; affects differentiation strategy.

4.4 Governance and trust as system requirements

Marketplace governance is not "policy only": it becomes a set of product and system requirements.

- Seller onboarding, verification, and compliance workflows (KYC/KYB where relevant).
- Catalog integrity: listing policies, moderation, and counterfeit detection.
- Reviews/ratings, dispute resolution, refunds/returns policy enforcement.
- Trust & safety operations: rule+ML detection, case management, and human-in-the-loop escalation.

4.5 Business model patterns (and architectural implications)

4.5.1 B2C retail

- Key systems: PIM/catalog, promotions, payments, fulfillment, customer support.
- ERP/CRM integration: order-to-cash, inventory, accounting, returns.

4.5.2 B2B commerce

- Quotes, negotiated pricing, approval workflows, invoicing, credit limits.
- Integration-heavy: procurement, supplier management, and EDI/API flows.

4.5.3 Marketplace

- Split catalog ownership (platform vs sellers), seller tools, payout/settlement.
- Additional "control plane": seller policies, listing moderation, abuse prevention.

4.5.4 Digital services (subscriptions)

- Recurring billing, entitlements, and churn/retention analytics.
- Usage-based pricing requires event design and metering.

4.6 Where AI fits (system-level view)

- **Matching layer:** search, recommendations, ranking.
- **Trust layer:** fraud/abuse detection, content moderation.
- **Operations layer:** forecasting, replenishment, routing.
- **Experience layer:** personalization and conversational commerce.

4.7 Hands-on (placeholder)

- Create a one-page *platform blueprint*: participant groups, value exchange, pricing model, and required capabilities.
- Map capabilities to an EA view and identify which capabilities are ERP-owned vs commerce-owned.

4.8 Case studies (placeholder)

- Modern marketplace patterns (generic; no vendor lock-in).

4.9 Multiple-choice questions (MCQs)

1. Which statement best distinguishes a *storefront* from a *marketplace*?
   (a) A storefront is always B2B; a marketplace is always B2C.
   (b) A storefront has a single merchant of record; a marketplace mediates transactions among multiple sellers and buyers.
   (c) A storefront cannot use AI; a marketplace must use AI.
   (d) A storefront requires ERP integration; a marketplace does not.
2. *Indirect network effects* in marketplaces most directly mean:
   (a) The platform has a single user group and growth is linear.
   (b) The value to buyers increases as more sellers join, and the value to sellers increases as more buyers join.
   (c) Users prefer multiple platforms (multi-homing) regardless of differentiation.
   (d) The platform must subsidize logistics to succeed.
3. In B2B commerce, which requirement is **most typical** compared to B2C?
   (a) Session-based recommendations
   (b) Negotiated pricing and approval workflows
   (c) Visual search
   (d) Social reviews and influencer marketing
4. *Multi-homing* refers to:
   (a) Hosting the platform in multiple cloud regions.
   (b) Sellers or buyers participating in multiple platforms simultaneously.
   (c) Using multiple payment gateways for redundancy.
   (d) Maintaining multiple warehouses for the same SKU.
5. Which is the **best** example of a marketplace *governance* capability?
   (a) Re-ranking items using embeddings
   (b) Seller onboarding verification and dispute resolution policies
   (c) Demand forecasting for replenishment
   (d) Offline model training with hyperparameter tuning

Answer key

1. (b)
2. (b)
3. (b)
4. (b)
5. (b)

Chapter 5

Platforms, Marketplaces, and Business Models (Week 2)

5.1 Learning objectives

- Distinguish e-commerce *channels* (storefronts) from *platforms* (ecosystems) and *marketplaces* (multi-seller).
- Analyze platform economics: network effects, multi-homing, and pricing of participation.
- Translate business model choices into architectural requirements (identity, trust, data, integration, ERP).

5.2 Multiple-choice questions (MCQs)

1. Which statement best distinguishes a *storefront* from a *marketplace*?
   (a) A storefront is always B2B; a marketplace is always B2C.
   (b) A storefront has a single merchant of record; a marketplace mediates transactions among multiple sellers and buyers.
   (c) A storefront cannot use AI; a marketplace must use AI.
   (d) A storefront requires ERP integration; a marketplace does not.
2. *Indirect network effects* in marketplaces most directly mean:
   (a) The platform has a single user group and growth is linear.
   (b) The value to buyers increases as more sellers join, and the value to sellers increases as more buyers join.
   (c) Users prefer multiple platforms (multi-homing) regardless of differentiation.
   (d) The platform must subsidize logistics to succeed.
3. In B2B commerce, which requirement is **most typical** compared to B2C?

(a) Session-based recommendations
(b) Negotiated pricing and approval workflows
(c) Visual search
(d) Social reviews and influencer marketing

4. *Multi-homing* refers to:

   (a) Hosting the platform in multiple cloud regions.
   (b) Sellers or buyers participating in multiple platforms simultaneously.
   (c) Using multiple payment gateways for redundancy.
   (d) Maintaining multiple warehouses for the same SKU.

5. Which is the **best** example of a marketplace *governance* capability?

   (a) Re-ranking items using embeddings
   (b) Seller onboarding verification and dispute resolution policies
   (c) Demand forecasting for replenishment
   (d) Offline model training with hyperparameter tuning

Answer key

1. (b)
2. (b)
3. (b)
4. (b)
5. (b)

5.3 Scope

- Multi-sided platforms, network effects, platform governance, and incentives.
- B2B e-commerce, procurement portals, and industry-specific marketplaces.

5.4 Core concepts for Master's students

5.4.1 From storefront to platform

- **Storefront:** a single merchant selling to buyers (B2C/B2B) via a web/app channel.
- **Platform:** a system that enables interactions among multiple participant groups (e.g., buyers, sellers, advertisers, logistics providers).
- **Marketplace:** a platform where multiple sellers list items and transactions are mediated by platform policies and infrastructure.

5.4.2 Value creation and capture

- **Value creation:** reduce search/transaction costs, increase variety, improve trust, and improve matching (often AI-driven).
- **Value capture:** take rate, subscription, listing fees, ads, fulfillment fees, and data/analytics products.

5.4.3 Network effects and growth loops

- **Direct network effects:** more users attract more users (e.g., social commerce communities).
- **Indirect network effects:** more buyers attract more sellers and vice versa (classic marketplace dynamic).
- **Cold start:** early-stage bootstrapping strategies (seed supply vs seed demand).
- **Multi-homing:** sellers/buyers participating in multiple platforms; implications for loyalty and differentiation.

5.4.4 Governance and trust

- Identity and onboarding (KYC/KYB where relevant), seller verification, and catalog integrity.
- Reviews/ratings, dispute management, returns policy, and service-level enforcement.
- Trust & safety as an *operating system* for the marketplace: policies + detection + human-in-the-loop.

5.5 Business model patterns (with architectural implications)

5.5.1 B2C retail

- Key systems: product information management (PIM), promotions, payments, fulfillment, customer service.
- ERP/CRM integration: order-to-cash, inventory, accounting, returns.

5.5.2 B2B commerce

- Quotation, contracts, negotiated pricing, credit limits, invoicing, and approval workflows.
- Integration-heavy by nature: procurement, supplier management, and EDI/API flows.

5.5.3 Marketplace

- Split catalog ownership (platform vs sellers), seller tools, and payout/settlement.
- Additional control plane: seller policies, listing moderation, and fraud/abuse prevention.

5.5.4 Digital services (subscriptions)

- Recurring billing, entitlement management, and churn/retention analytics.
- Usage-based pricing requires event design and metering.

5.6 AI is not "a feature": where it fits in the platform

- **Matching layer:** search, recommendations, and ranking across buyers and sellers.
- **Trust layer:** fraud detection, abuse detection, and content moderation.
- **Operations layer:** forecasting, replenishment, routing, and workforce planning.
- **Experience layer:** conversational commerce, assistants, and personalization.

5.7 Hands-on (placeholder)

- Build a one-page *platform blueprint*: participant groups, value exchange, pricing model, and required capabilities.
- Map the blueprint to a capability map and identify which capabilities are ERP-owned vs commerce-owned.

5.8 Core concepts for Master's students

5.8.1 From storefront to platform

- **Storefront:** a single merchant selling to buyers (B2C/B2B) via a web/app channel.
- **Platform:** a system that enables interactions among multiple participant groups (e.g., buyers, sellers, advertisers, logistics providers).
- **Marketplace:** a platform where multiple sellers list items and transactions are mediated by platform policies and infrastructure.

5.8.2 Value creation and capture

- **Value creation:** reduce search/transaction costs, increase variety, improve trust, and improve matching (often AI-driven).
- **Value capture:** take rate, subscription, listing fees, ads, fulfillment fees, and data/analytics products.

5.8.3 Network effects and growth loops

- **Direct network effects:** more users attract more users (e.g., social commerce communities).
- **Indirect network effects:** more buyers attract more sellers and vice versa (classic marketplace dynamic).
- **Cold start:** early-stage bootstrapping strategies (seed supply vs seed demand).
- **Multi-homing:** sellers/buyers participating in multiple platforms; implications for loyalty and differentiation.

5.8.4 Governance and trust

- Identity and onboarding (KYC/KYB where relevant), seller verification, and catalog integrity.
- Reviews/ratings, dispute management, returns policy, and service-level enforcement.
- Trust & safety as an *operating system* for the marketplace: policies + detection + human-in-the-loop.

5.9 Business model patterns (with architectural implications)

5.9.1 B2C retail

- Key systems: product information management (PIM), promotions, payments, fulfillment, customer service.
- ERP/CRM integration: order-to-cash, inventory, accounting, returns.

5.9.2 B2B commerce

- Quotation, contracts, negotiated pricing, credit limits, invoicing, and approval workflows.
- Integration-heavy by nature: procurement, supplier management, and EDI/API flows.

5.9.3 Marketplace

- Split catalog ownership (platform vs sellers), seller tools, and payout/settlement.
- Additional control plane: seller policies, listing moderation, and fraud/abuse prevention.

5.9.4 Digital services (subscriptions)

- Recurring billing, entitlement management, and churn/retention analytics.
- Usage-based pricing requires event design and metering.

5.10 AI is not "a feature": where it fits in the platform

- **Matching layer:** search, recommendations, and ranking across buyers and sellers.
- **Trust layer:** fraud detection, abuse detection, and content moderation.
- **Operations layer:** forecasting, replenishment, routing, and workforce planning.
- **Experience layer:** conversational commerce, assistants, and personalization.

5.11 Hands-on (placeholder)

- Build a one-page *platform blueprint*: participant groups, value exchange, pricing model, and required capabilities.
- Map the blueprint to a capability map and identify which capabilities are ERP-owned vs commerce-owned.

5.12 Case studies (placeholder)

- Modern marketplace patterns (generic, no vendor lock-in)

5.13 Multiple-choice questions (MCQs)

1. Which statement best distinguishes a *storefront* from a *marketplace*?
   (a) A storefront is always B2B; a marketplace is always B2C.
   (b) A storefront has a single merchant of record; a marketplace mediates transactions among multiple sellers and buyers.
   (c) A storefront cannot use AI; a marketplace must use AI.
   (d) A storefront requires ERP integration; a marketplace does not.
2. *Indirect network effects* in marketplaces most directly mean:
   (a) The platform has a single user group and growth is linear.
   (b) The value to buyers increases as more sellers join, and the value to sellers increases as more buyers join.
   (c) Users prefer multiple platforms (multi-homing) regardless of differentiation.
   (d) The platform must subsidize logistics to succeed.
3. In B2B commerce, which requirement is **most typical** compared to B2C?
   (a) Session-based recommendations
   (b) Negotiated pricing and approval workflows
   (c) Visual search

   (d) Social reviews and influencer marketing

4. *Multi-homing* refers to:
   (a) Hosting the platform in multiple cloud regions.
   (b) Sellers or buyers participating in multiple platforms simultaneously.
   (c) Using multiple payment gateways for redundancy.
   (d) Maintaining multiple warehouses for the same SKU.

5. Which is the **best** example of a marketplace *governance* capability?
   (a) Re-ranking items using embeddings
   (b) Seller onboarding verification and dispute resolution policies
   (c) Demand forecasting for replenishment
   (d) Offline model training with hyperparameter tuning

Answer key

1. (b)
2. (b)
3. (b)
4. (b)
5. (b)

Chapter 6

E-Commerce Data Foundations: Events, Experimentation, and Metrics (Week 3)

6.1 Scope (placeholder)

- Event schema design, identity resolution, attribution caveats, and data quality.
- A/B testing, bandits, and experimentation at scale (conceptual).

6.2 Hands-on lab (placeholder)

- Build an event log + simple metrics layer for conversion and retention.

6.3 Multiple-choice questions (MCQs)

1. In event-based analytics for e-commerce, which statement is most accurate?
   (a) Events should be stored without timestamps to save space.
   (b) Events are most useful when they have a consistent schema, a timestamp, and stable identifiers.
   (c) Only purchase events matter; click and view events are noise.
   (d) The best schema changes frequently to match UI updates.
2. Which is an example of a *data quality* problem that can directly harm ML models?
   (a) A model uses a neural network instead of linear regression.
   (b) Missing or duplicated events for key actions (e.g., add-to-cart) create biased labels/features.
   (c) Using SQL instead of Python for ETL.

(d) Choosing a smaller batch size during training.

3. Why are offline metrics often insufficient for evaluating an e-commerce ranking model?
   (a) Offline metrics are always higher than online metrics.
   (b) Offline metrics cannot measure causal impact under real user behavior and feedback loops.
   (c) Offline evaluation cannot be computed from logs.
   (d) Offline metrics are not used in industry.
4. In an A/B test, the primary purpose of randomization is to:
   (a) Increase model complexity.
   (b) Ensure the treatment and control groups are comparable in expectation.
   (c) Eliminate the need for monitoring.
   (d) Guarantee the KPI improves.
5. A common *identity resolution* challenge in e-commerce is:
   (a) Mapping product IDs to SKU IDs.
   (b) Linking the same person across devices/sessions while respecting privacy constraints.
   (c) Computing NDCG@k efficiently.
   (d) Selecting a cloud region.

Answer key

1. (b)
2. (b)
3. (b)
4. (b)
5. (b)

6.4 Exercises (short)

1. Propose an event taxonomy for a storefront: list at least 10 events (search, view, add-to-cart, checkout steps, purchase, return) and for each event specify 3–5 key fields.
2. Define a North Star metric for an e-commerce product and break it into 3–5 leading indicators. Explain how you would compute each from event logs.
3. Design a simple A/B test for a recommendation widget. Specify: unit of randomization, primary KPI, guardrail metrics, and an example of a bias/confounder to watch for.

6.5 Mini-case (Odoo-linked, vendor-neutral)

Scenario: A hybrid B2C/B2B company uses Odoo (Sales, Inventory, Accounting, CRM). The storefront and mobile app generate clickstream events.

Task: Design a minimal data model that supports both analytics and ML:

- Define how you will link clickstream identities to Odoo customers/partners (when possible) and how you handle guests.
- Propose which entities should have stable IDs across systems (customer, product, order, shipment) and which are channel-specific.
- Specify 5 "data contracts" (producer → consumer) that reduce breakage when the UI or ERP changes.

Chapter 7

Enterprise Architecture for E-Commerce (Week 4)

7.1 Scope (placeholder)

- Capability maps, value streams, domain boundaries, and target architecture.
- Reference views: business, application, data, and technology architectures.

7.2 Artifacts (placeholder)

- Deliverable: e-commerce capability map + domain model.

7.3 Multiple-choice questions (MCQs)

1. In enterprise architecture, a *capability map* primarily describes:
   (a) A list of specific microservices and their endpoints.
   (b) *What* the business does (stable abilities), independent of *how* it is implemented.
   (c) The physical network topology of the data center.
   (d) A Gantt chart of the project plan.
2. A good domain boundary (e.g., for a "Catalog" domain) typically aims to:
   (a) Maximize shared database tables across all teams.
   (b) Minimize coupling and define clear ownership of data and behaviors.
   (c) Ensure all operations are synchronous.
   (d) Avoid having APIs.
3. Which artifact is most suitable for describing how business value is delivered end-to-end?

(a) Value stream map
(b) Entity-relationship diagram (ERD)
(c) Source code repository layout
(d) TLS configuration file

4. "Target architecture" is best described as:

(a) The current-state system diagram.
(b) A future-state design that guides decisions and sequencing of change.
(c) A vendor product brochure.
(d) A test plan for unit tests.

5. In an AI-driven e-commerce context, which is the best example of an *architectural concern* (not a model choice)?

(a) Whether to use XGBoost or logistic regression
(b) How to ensure training-serving consistency and monitor drift
(c) Whether to use k-means or DBSCAN
(d) Whether to use SGD or Adam

Answer key

1. (b)
2. (b)
3. (a)
4. (b)
5. (b)

7.4 Exercises (short)

1. Draft a capability map for an AI-enabled e-commerce organization. Include at least: customer acquisition, discovery, pricing/promotions, order management, fulfillment/returns, customer support, data/analytics, and governance/security.

2. Choose one value stream (e.g., "order-to-cash" or "browse-to-buy"). Identify 5–8 steps and list the owning domain/system for each step (commerce vs ERP vs logistics vs payment).

3. Propose domain boundaries for: catalog, pricing, orders, payments, and customer profiles. For each boundary, name the system-of-record for key data entities.

7.5 Mini-case (Odoo-linked, vendor-neutral)

Scenario: Your company is hybrid B2C/B2B. Odoo is used for core ERP flows (Sales, Inventory, Accounting, CRM). A separate commerce layer handles web/mobile experiences and AI features.

Task: Produce a short "target architecture" note:

- Draw a capability-to-system mapping (capabilities → commerce layer vs Odoo vs data platform).
- Define the *system of record* for customer, product, price list, order, invoice, and shipment.
- Identify 3 integration risks (data consistency, latency, duplicate sources of truth) and propose mitigations.

Chapter 8

Enterprise Integration Patterns for E-Commerce (Week 5)

8.1 Scope (placeholder)

- API-first, integration styles (sync/async), event-driven architecture.
- iPaaS/ESB concepts, message brokers/streams, and reliability patterns.

8.2 Hands-on lab (placeholder)

- Model an order lifecycle with events + idempotency + outbox pattern (conceptual + pseudo).

8.3 Multiple-choice questions (MCQs)

1. Which statement best characterizes synchronous vs asynchronous integration?
   (a) Synchronous integration cannot fail; asynchronous always fails.
   (b) Synchronous integration couples availability and latency; asynchronous trades immediacy for resilience and decoupling.
   (c) Asynchronous integration is only for analytics; synchronous is only for ERP.
   (d) They are equivalent as long as JSON is used.
2. Why is idempotency important in order processing APIs?
   (a) It makes responses larger.
   (b) It prevents duplicate effects when requests are retried.
   (c) It eliminates the need for authentication.
   (d) It guarantees exactly-once message delivery in all systems.
3. The *outbox pattern* is primarily used to:

(a) Store images for the catalog.
(b) Reliably publish events when database writes succeed, avoiding dual-write inconsistencies.
(c) Encrypt API traffic.
(d) Replace a message broker.

4. In event-driven systems, a *consumer* should typically be designed to be:

   (a) State-less but non-retryable
   (b) Idempotent and retryable
   (c) Dependent on message ordering always being perfect
   (d) Tightly coupled to producer database schemas

5. Which is a common reason to choose events over direct API calls for some flows?

   (a) To avoid defining schemas/contracts
   (b) To decouple systems and allow multiple downstream consumers without changing the producer
   (c) To guarantee zero latency
   (d) To avoid monitoring

Answer key

1. (b)
2. (b)
3. (b)
4. (b)
5. (b)

8.4 Exercises (short)

1. For each flow, choose sync API or async events and justify: (i) "place order", (ii) "update inventory availability", (iii) "invoice posted", (iv) "shipment delivered", (v) "customer profile updated".

2. Design an idempotency strategy for the "create order" API. Specify the idempotency key and how long it is retained.

3. Write a short failure story: a dual-write bug between database and message broker. Explain how the outbox pattern prevents it.

8.5 Mini-case (Odoo-linked, vendor-neutral)

Scenario: A headless storefront integrates with Odoo for orders, inventory, invoices, and CRM updates.

Task: Propose an integration design that supports scale and reliability:

- Define the key domain events (at least 8) and which system publishes each.
- Identify where you need synchronous APIs (e.g., payment authorization) vs asynchronous events (e.g., shipment updates).
- Specify two data contracts and a versioning strategy to avoid breaking changes.
- List the minimum observability signals (logs/metrics/traces) required to operate the integration in production.

8.6 Multiple-choice questions (MCQs)

1. Which statement best characterizes synchronous vs asynchronous integration?
   (a) Synchronous integration cannot fail; asynchronous always fails.
   (b) Synchronous integration couples availability and latency; asynchronous trades immediacy for resilience and decoupling.
   (c) Asynchronous integration is only for analytics; synchronous is only for ERP.
   (d) They are equivalent as long as JSON is used.
2. Why is idempotency important in order processing APIs?
   (a) It makes responses larger.
   (b) It prevents duplicate effects when requests are retried.
   (c) It eliminates the need for authentication.
   (d) It guarantees exactly-once message delivery in all systems.
3. The *outbox pattern* is primarily used to:
   (a) Store images for the catalog.
   (b) Reliably publish events when database writes succeed, avoiding dual-write inconsistencies.
   (c) Encrypt API traffic.
   (d) Replace a message broker.
4. In event-driven systems, a *consumer* should typically be designed to be:
   (a) State-less but non-retryable
   (b) Idempotent and retryable
   (c) Dependent on message ordering always being perfect

   (d) Tightly coupled to producer database schemas

5. Which is a common reason to choose events over direct API calls for some flows?
   (a) To avoid defining schemas/contracts
   (b) To decouple systems and allow multiple downstream consumers without changing the producer
   (c) To guarantee zero latency
   (d) To avoid monitoring

Answer key

1. (b)
2. (b)
3. (b)
4. (b)
5. (b)

8.7 Exercises (short)

1. For each flow, choose sync API or async events and justify: (i) "place order", (ii) "update inventory availability", (iii) "invoice posted", (iv) "shipment delivered", (v) "customer profile updated".
2. Design an idempotency strategy for the "create order" API. Specify the idempotency key and how long it is retained.
3. Write a short failure story: a dual-write bug between database and message broker. Explain how the outbox pattern prevents it.

8.8 Mini-case (Odoo-linked, vendor-neutral)

Scenario: A headless storefront integrates with Odoo for orders, inventory, invoices, and CRM updates.

Task: Propose an integration design that supports scale and reliability:

- Define the key domain events (at least 8) and which system publishes each.
- Identify where you need synchronous APIs (e.g., payment authorization) vs asynchronous events (e.g., shipment updates).
- Specify two data contracts and a versioning strategy to avoid breaking changes.

- List the minimum observability signals (logs/metrics/traces) required to operate the integration in production.

Chapter 9

ERP/CRM/SCM Integration in Commerce (Week 6)

9.1 Scope (placeholder)

- Odoo overview (high-level): key modules relevant to commerce (Sales, Inventory, Accounting, CRM).
- Order-to-cash, inventory, fulfillment, returns, and customer data flows.
- Master data management (MDM) and data ownership across domains.

9.2 Artifacts (placeholder)

- Deliverable: integration context diagram + data contracts (high level), including an Odoo boundary.

9.3 Multiple-choice questions (MCQs)

1. In an integrated commerce + ERP architecture, the *system of record* is best defined as:
   (a) The system with the nicest UI
   (b) The system that is most convenient for analytics
   (c) The authoritative source of truth for a given entity (e.g., invoice) and its lifecycle
   (d) The system that runs in the cloud
2. Which flow is most strongly associated with *order-to-cash*?
   (a) Creating a product image embedding
   (b) Quote/order $\rightarrow$ delivery $\rightarrow$ invoice $\rightarrow$ payment reconciliation
   (c) Building a recommendation model

   (d) Defining a Kafka topic

3. Master data management (MDM) is primarily concerned with:
   (a) Training deep learning models for search
   (b) Defining and governing shared entities (customer, product, supplier) and preventing duplicates/inconsistencies
   (c) Choosing the best database index
   (d) Encrypting API requests

4. A common integration risk when connecting commerce with ERP is:
   (a) Too many unit tests
   (b) Duplicate sources of truth for prices, customers, or inventory
   (c) Having an event schema
   (d) Using idempotency keys

5. In practice, why do integrations often require both APIs and events?
   (a) Because events are always cheaper than APIs
   (b) Because some steps require immediate confirmation while others benefit from decoupling and retries
   (c) Because ERP systems cannot expose APIs
   (d) Because events guarantee zero duplicates without design effort

Answer key

1. (c)
2. (b)
3. (b)
4. (b)
5. (b)

9.4 Exercises (short)

1. For each entity, choose a system of record (commerce vs Odoo) and justify: customer, product, price list, inventory on-hand, order, invoice, payment, return.
2. Describe two failure modes of inventory sync and propose mitigations (e.g., eventual consistency, reservations, reconciliation jobs).
3. Draft a "data contract" for an `OrderCreated` event: required fields, optional fields, and versioning rules.

9.5 Mini-case (Odoo-linked, vendor-neutral)

Scenario: A hybrid company runs a headless storefront for B2C and a B2B portal for wholesale customers. Odoo is used for Sales, Inventory, Accounting, and CRM.

Task: Design the integration for three end-to-end processes:

- **Order-to-cash:** from checkout to invoice posting and payment reconciliation.
- **Returns/refunds:** from return request to stock adjustment and credit note.
- **B2B pricing:** negotiated price lists and approvals without duplicating sources of truth.

For each process, state: key events, required synchronous calls, idempotency strategy, and the minimum monitoring you would implement.

9.6 Multiple-choice questions (MCQs)

1. In an integrated commerce + ERP architecture, the *system of record* is best defined as:
   (a) The system with the nicest UI
   (b) The system that is most convenient for analytics
   (c) The authoritative source of truth for a given entity (e.g., invoice) and its lifecycle
   (d) The system that runs in the cloud
2. Which flow is most strongly associated with *order-to-cash*?
   (a) Creating a product image embedding
   (b) Quote/order → delivery → invoice → payment reconciliation
   (c) Building a recommendation model
   (d) Defining a Kafka topic
3. Master data management (MDM) is primarily concerned with:
   (a) Training deep learning models for search
   (b) Defining and governing shared entities (customer, product, supplier) and preventing duplicates/inconsistencies
   (c) Choosing the best database index
   (d) Encrypting API requests
4. A common integration risk when connecting commerce with ERP is:
   (a) Too many unit tests
   (b) Duplicate sources of truth for prices, customers, or inventory
   (c) Having an event schema
   (d) Using idempotency keys

5. In practice, why do integrations often require both APIs and events?
   (a) Because events are always cheaper than APIs
   (b) Because some steps require immediate confirmation while others benefit from decoupling and retries
   (c) Because ERP systems cannot expose APIs
   (d) Because events guarantee zero duplicates without design effort

Answer key

1. (c)
2. (b)
3. (b)
4. (b)
5. (b)

9.7 Exercises (short)

1. For each entity, choose a system of record (commerce vs Odoo) and justify: customer, product, price list, inventory on-hand, order, invoice, payment, return.
2. Describe two failure modes of inventory sync and propose mitigations (e.g., eventual consistency, reservations, reconciliation jobs).
3. Draft a "data contract" for an `OrderCreated` event: required fields, optional fields, and versioning rules.

9.8 Mini-case (Odoo-linked, vendor-neutral)

Scenario: A hybrid company runs a headless storefront for B2C and a B2B portal for wholesale customers. Odoo is used for Sales, Inventory, Accounting, and CRM.

Task: Design the integration for three end-to-end processes:

- **Order-to-cash:** from checkout to invoice posting and payment reconciliation.
- **Returns/refunds:** from return request to stock adjustment and credit note.
- **B2B pricing:** negotiated price lists and approvals without duplicating sources of truth.

For each process, state: key events, required synchronous calls, idempotency strategy, and the minimum monitoring you would implement.

Part III

Machine Learning and Deep Learning for E-Commerce

Chapter 10

Recommenders I: Classical Methods + Learning-to-Rank (Week 7)

10.1 Scope (placeholder)

- Collaborative filtering, matrix factorization, and implicit feedback.
- Learning-to-rank: pointwise/pairwise/listwise framing; offline vs online evaluation.

10.2 Hands-on lab (placeholder)

- Baseline recommender with clear metrics (MAP@K/NDCG@K).

Chapter 11

Recommenders II: Deep Learning, Embeddings, and Retrieval (Week 8)

11.1 Scope (placeholder)

- Two-tower retrieval, embeddings, sequence models, and candidate generation vs re-ranking.
- Practical concerns: cold-start, diversity, bias, and latency.

11.2 Hands-on lab (placeholder)

- Train embeddings; build a retrieval + re-ranking pipeline (high level).

Chapter 12

Pricing and Promotions with ML + Optimization (Week 9)

12.1 Scope (placeholder)

- Demand forecasting, elasticity, uplift, and constraints (inventory/competition).
- Optimization approaches and experiment design for pricing.

12.2 Hands-on lab (placeholder)

- Forecast baseline + simple constrained optimization scenario.

Chapter 13

Fraud, Risk, and Trust & Safety (Week 10)

13.1 Scope (placeholder)

- Fraud types: payment, account takeover, promotion abuse, refund abuse.
- Methods: supervised learning, anomaly detection, graph ML, and rule+ML hybrids.

13.2 Hands-on lab (placeholder)

- Build a risk scoring baseline + thresholding and cost-sensitive evaluation.

Part IV

Applied AI Systems, LLMs, and Operations

Chapter 14

Customer Service Automation with LLMs (Week 11)

14.1 Scope (placeholder)

- RAG, tool use, workflow orchestration, evaluation, and guardrails.
- When to use LLMs vs classical NLP vs rules.

14.2 Hands-on lab (placeholder)

- Prototype: FAQ assistant with retrieval + citations (local documents only).

Chapter 15

MLOps + DataOps for E-Commerce (Week 12)

15.1 Scope (placeholder)

- Platform-agnostic MLOps/DataOps: CI/CD for ML, pipelines, and reproducibility.
- Feature stores (concept), training-serving skew, monitoring, drift, and SLOs.
- Experiment tracking, model governance, and evaluation at scale.

15.2 Artifacts (placeholder)

- Deliverable: ML system design doc + monitoring plan.

Chapter 16

Security, Privacy, Compliance, and Responsible AI (Week 13)

16.1 Scope (placeholder)

- Payments/security basics, privacy-by-design, and data minimization.
- Threat modeling for e-commerce and AI features (LLM-specific risks included).
- Responsible AI: fairness, explainability, auditability, and human-in-the-loop controls.

16.2 Discussion (placeholder)

- Risk assessment template for an AI feature.

Chapter 17

Capstone: Enterprise-Grade AI E-Commerce Architecture (Week 14)

Capstone scenario (agreed)

- Scenario: end-to-end "AI commerce + Odoo" reference architecture for a single company.
- Goal: demonstrate EA alignment, ERP integration, and measurable AI value with operational readiness.
- Business model: hybrid (B2C + B2B).
- Commerce front-end: both (compare Odoo Website/eCommerce vs a separate headless storefront integrating to Odoo).
- AI stack: both (ML for recs/fraud/pricing + LLMs for support/ops).

17.1 Scope (placeholder)

- End-to-end architecture: channels, services, data platform, ML/LLM layer, ERP integration.
- Evaluation: business metrics, technical metrics, and operational readiness.

17.2 Capstone deliverables (placeholder)

- Architecture pack (diagrams), backlog, data contracts, and evaluation plan.

Part V

Appendices (Placeholders)

Appendix A

Datasets and Synthetic Data for Teaching (placeholder)

Appendix B

Template: System Design Document (placeholder)

Appendix C

Template: Data Contract (placeholder)

Appendix D

Template: Model Card and Risk Assessment (placeholder)