AI PRODUCT MANAGEMENT 2 · DETAILED COURSE

Technical Foundations for Product Managers

Enough software and systems knowledge to discuss trade-offs credibly with engineers and make sound decisions.

Chapter 10 of 26 · 28 topics · Edition 2026 · skillnavora.in

CHAPTER 10

Technical Foundations for Product Managers

Enough software and systems knowledge to discuss trade-offs credibly with engineers and make sound decisions.

How Software Applications Work

Users interact with a front end that calls back-end services, which read and write data and return results.

A user interacts with a front end, which sends requests to back-end services. Those services apply business rules, read and write data and return results. Knowing this flow helps a PM describe where a bug or delay happens.

Software Development Lifecycle (SDLC)

Plan, design, build, test, deploy and maintain, in sequential or iterative form.

The SDLC covers planning, design, build, test, deploy and maintain. It can run in sequence or iteratively. Each stage has outputs such as requirements, designs, code and release notes that the PM reviews.

Frontend Fundamentals

HTML, CSS and JavaScript in the browser or native code on mobile, handling what users see and touch.

Frontend code (HTML, CSS and JavaScript on the web, or native code on mobile) controls what users see and touch. Performance on frontend depends on page weight, rendering and network requests.

Backend Fundamentals

Server logic, business rules, databases and integrations that power the front end.

The back end runs server logic, business rules, databases and integrations. Latency and reliability problems often start here, so PMs should understand what runs on the back end and what each request costs.

Client-Server Architecture

Clients send requests; servers process and respond, over a network.

Clients send requests and servers process them and send responses over a network. Most apps follow this pattern. Consider where data is processed when designing offline or low-bandwidth features.

Databases and Data Storage

Persistent stores for application data, chosen for structure, scale and query needs.

Databases persist application data. The choice depends on structure, scale, consistency and query needs. Changing a database later is expensive, so involve engineers early in data decisions.

SQL and NoSQL Databases

SQL stores structured tables with strong consistency; NoSQL covers document, key-value, column and graph stores built for flexibility and scale.

SQL databases store structured tables with strong consistency and joins. NoSQL covers document, key-value, column and graph stores that trade some consistency or queryability for flexibility and scale.

Relational Database Concepts

Tables, keys, relationships, normalisation and transactions (ACID).

Relational databases use tables, keys, relationships and normalisation. Transactions follow ACID properties (atomicity, consistency, isolation, durability), which keep money movements correct.

APIs and API Fundamentals

Contracts that let software components talk, with defined requests, responses and rules.

An API is a contract that lets software components communicate. It defines requests, responses, authentication and error behaviour. Good APIs are stable and documented, because other teams depend on them.

REST APIs

Resource-based HTTP APIs using verbs like GET, POST, PUT and DELETE and status codes.

REST exposes resources through HTTP verbs such as GET, POST, PUT and DELETE, and uses status codes to report results. Example: GET /orders/123 returns one order.

GraphQL

A query language where clients request exactly the fields they need from one endpoint.

GraphQL lets clients request exactly the fields they need from a single endpoint. It reduces over-fetching for complex screens but needs care with query cost and caching.

Webhooks

Event-driven callbacks where one system sends an HTTP request to another when something happens.

A webhook sends an HTTP request from one system to another when an event occurs, such as payment succeeded. It avoids constant polling, but the receiver must handle retries and duplicate events.

Authentication and Authorisation

Authentication proves who you are; authorisation decides what you may do (OAuth, SSO, roles).

Authentication proves identity, for example with a password or single sign-on. Authorisation decides what that identity may do, using roles or permissions. OAuth is a common standard for delegated access.

JSON and XML

Text formats for exchanging structured data; JSON is lighter and more common today.

JSON and XML are text formats for structured data. JSON is lighter and standard for web APIs. Knowing the format helps a PM read API responses and specifications.

API Documentation

Reference, guides and examples (often OpenAPI) that make APIs easy to adopt.

Good documentation includes reference pages, guides and examples, often generated from an OpenAPI specification. Clear docs cut support requests and speed up integration.

Postman

A tool to explore, test and document APIs by sending requests and viewing responses.

Postman lets you send requests, inspect responses and save collections of API calls. PMs use it to understand how an API behaves before writing requirements.

Git and GitHub Fundamentals

Git tracks changes; GitHub hosts repositories and supports pull requests and reviews.

Git tracks changes to code. GitHub hosts repositories and supports pull requests and code review. Knowing the basic workflow lets a PM follow release discussions and documentation changes.

Version Control

Recording history of changes so teams can collaborate, branch, review and roll back.

Version control records history so teams can collaborate, branch, review and roll back. Rollbacks depend on it, and it is the audit trail for what changed in each release.

Microservices Architecture

Many small, independently deployable services that communicate over APIs.

Microservices split an application into small, independently deployable services that talk over APIs. They help large teams work in parallel, but add operational complexity and network failure modes.

Monolithic Architecture

One deployable unit containing all functionality: simple to start, harder to scale teams.

A monolith is one deployable unit containing all functionality. It is simpler to start and debug, but large teams can slow each other down. Many successful products begin as monoliths.

System Design Fundamentals

Choosing components, data flow and trade-offs to meet requirements for scale, cost and reliability.

System design chooses components, data flows and trade-offs to meet requirements for scale, cost and reliability. PMs do not design systems alone, but they should know which trade-offs their requirements create.

Scalability and Reliability

Handling growth in load and staying available despite failures, using redundancy, caching and graceful degradation.

Scalability handles growing load, and reliability keeps the service available despite failures. Techniques include redundancy, caching, load balancing and graceful degradation, such as showing cached data when a service is slow.

Latency and Throughput

Latency is time per request; throughput is requests handled per unit time.

Latency is the time for one request. Throughput is the number of requests handled per unit time. A system can have high throughput and poor latency, so track both.

Cloud Computing Fundamentals

On-demand computing, storage and services paid by use: IaaS, PaaS, SaaS, regions and shared responsibility.

Cloud services provide compute, storage and managed services on demand, paid by use. Models include IaaS, PaaS and SaaS. Responsibility for security is shared between provider and customer.

AWS Fundamentals

Amazon's cloud: EC2, S3, Lambda, RDS and IAM as core building blocks.

AWS offers services such as EC2 for virtual machines, S3 for storage, Lambda for functions, RDS for managed databases and IAM for access control. Knowing these names helps in technical discussions.

Microsoft Azure Fundamentals

Microsoft's cloud: virtual machines, storage, Azure AD, Azure OpenAI and DevOps.

Azure provides virtual machines, storage, Azure Active Directory for identity, Azure OpenAI for model access and DevOps tools. Many enterprises already use Azure, which affects procurement and integration choices.

Google Cloud Platform Fundamentals

Google's cloud: Compute Engine, Cloud Storage, BigQuery and Vertex AI.

GCP offers Compute Engine, Cloud Storage, BigQuery for analytics and Vertex AI for machine learning. Its data and AI services are often chosen for analytics-heavy products.

Cybersecurity Fundamentals

Confidentiality, integrity, availability, encryption, secure coding, access control and common attacks.

Security covers confidentiality, integrity and availability, plus encryption, secure coding, access control and common attacks. A PM must treat security requirements as product requirements, not as later clean-up.

Summary of terms

Term	Meaning
How Software Applications Work	Users interact with a front end that calls back-end services, which read and write data and return results.
Software Development Lifecycle (SDLC)	Plan, design, build, test, deploy and maintain, in sequential or iterative form.
Frontend Fundamentals	HTML, CSS and JavaScript in the browser or native code on mobile, handling what users see and touch.
Backend Fundamentals	Server logic, business rules, databases and integrations that power the front end.
Client-Server Architecture	Clients send requests; servers process and respond, over a network.
Databases and Data Storage	Persistent stores for application data, chosen for structure, scale and query needs.
SQL and NoSQL Databases	SQL stores structured tables with strong consistency; NoSQL covers document, key-value, column and graph stores built for flexibility and scale.
Relational Database Concepts	Tables, keys, relationships, normalisation and transactions (ACID).
APIs and API Fundamentals	Contracts that let software components talk, with defined requests, responses and rules.
REST APIs	Resource-based HTTP APIs using verbs like GET, POST, PUT and DELETE and status codes.
GraphQL	A query language where clients request exactly the fields they need from one endpoint.
Webhooks	Event-driven callbacks where one system sends an HTTP request to another when something happens.
Authentication and Authorisation	Authentication proves who you are; authorisation decides what you may do (OAuth, SSO, roles).
JSON and XML	Text formats for exchanging structured data; JSON is lighter and more common today.
API Documentation	Reference, guides and examples (often OpenAPI) that make APIs easy to adopt.
Postman	A tool to explore, test and document APIs by sending requests and viewing responses.
Git and GitHub Fundamentals	Git tracks changes; GitHub hosts repositories and supports pull requests and reviews.
Version Control	Recording history of changes so teams can collaborate, branch, review and roll back.
Microservices Architecture	Many small, independently deployable services that communicate over APIs.
Monolithic Architecture	One deployable unit containing all functionality: simple to start, harder to scale teams.
System Design Fundamentals	Choosing components, data flow and trade-offs to meet requirements for scale, cost and reliability.
Scalability and Reliability	Handling growth in load and staying available despite failures, using redundancy, caching and graceful degradation.

Latency and Throughput	Latency is time per request; throughput is requests handled per unit time.
Cloud Computing Fundamentals	On-demand computing, storage and services paid by use: IaaS, PaaS, SaaS, regions and shared responsibility.
AWS Fundamentals	Amazon's cloud: EC2, S3, Lambda, RDS and IAM as core building blocks.
Microsoft Azure Fundamentals	Microsoft's cloud: virtual machines, storage, Azure AD, Azure OpenAI and DevOps.
Google Cloud Platform Fundamentals	Google's cloud: Compute Engine, Cloud Storage, BigQuery and Vertex AI.
Cybersecurity Fundamentals	Confidentiality, integrity, availability, encryption, secure coding, access control and common attacks.

Practice exercise: Call a public REST API in Postman, read the JSON response, and draw the client, server and database flow for that request.

Self-check

1. Explain How Software Applications Work in two sentences to a non-technical stakeholder, then say what you would measure for it.
2. Explain API Documentation in two sentences to a non-technical stakeholder, then say what you would measure for it.
3. Explain Cybersecurity Fundamentals in two sentences to a non-technical stakeholder, then say what you would measure for it.

Related SkillNavora material

- System Design and Databases for PMs : Architecture, databases and trade-offs.
- Platform and API Product Management : API products, versioning and developer experience.
- AI Tutor : Ask questions on any topic.
- Study Library : Every lesson, class and PDF by programme.

© SkillNavora · skillnavora.in · Facts and tools change; check official sites before relying on a figure.

ORIGINAL CONTENT

Original AI Product Management content

The AI Product Management programme already on SkillNavora, included unchanged so the full course is in one place.

Quick links Study Library (all material) Learning sites for PM AI builder tools (how-to) Webhooks & Playwright MCP Webhook URLs and keys (start here) AI Skills Academy AI Skills Map Context engineering (PDF) AI skills & templates (PDF) Orchestration (PDF) PRD library (120 ideas) AI PRD Writer AI PM Toolkit AI Evals (PDF) Browser automation (PDF) AI second brain (PDF) PM handbook Top 100 PM questions AI product sense Q&A (86) PM practice test Practice Lab Tool tutorials AI Agents & MCP for PMs

PRD documents

PRD library: template, framing guide and 120 PRD ideas

Learn how top PMs frame a PRD, download the free template, and turn any of 120 ideas across 15 industries into a full PRD with AI.

Open the PRD library ✨ AI PRD Writer PRD template (Word) AI Evals for PMs (PDF)

Context window

Context window: what a model can see at once

The context window is the most tokens a model can consider in one request. It covers your prompt, any documents you paste, the conversation so far and the reply. Anything outside it is invisible to the model. For PMs: a longer window fits more material but costs more and can slow responses; put the most important facts near the instructions; use retrieval (RAG) rather than pasting whole libraries; and test answers on long inputs, where quality can drop.

Context engineering course Context windows in the AI PM 2 course

Tokens and context in the Grokbot course

AI-First Product Management

Resources for Product Managers

The PM Master Handbook, AI and product tools with their official websites, interview practice, a frameworks cheat sheet, a PRD template and free courses.

Open the PM handbook library: all 68 PDFs ↗ PM Glossary Handbook Tokens, Pricing and Tokenization Vibe Coding addendum

PM Master Handbook by SkillNavora
23 modules and 620+ pages: 12 Product Core modules, an 11-class AI Tools Lab, frameworks, case studies, projects and interview prep.

Open the interactive handbook ↗ Complete handbook PDF (620+ pages, 23 MB) ↗

Product Core modules

1Fundamentals of Product Management*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
2Product Strategy and Strategic Thinking*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
3Problem Discovery and Market Research*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
4Problem Framing and Prioritization*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
5UX and Design Collaboration*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
6Effective Communication and Stakeholder Management*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
7Product Analytics and Metrics*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
8Product Launch, Growth, and Monetization*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
9Tech 101 and System Design for PMs*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
10LLMs, RAGs, and AI Foundations*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
11Vibe Coding for Product Managers*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides
12Agentic AI and AI Systems*Handbook chapter, detailed PDF, handbook and class slides*PDF Detailed PDF Handbook Class Slides

AI Tools Lab

13AI Tools Landscape and Your Starter StackPDF ↗ 14Claude for Daily PM WorkPDF ↗ 15Claude Code: Basic to AdvancedPDF ↗ 16Lovable: Basic to AdvancedPDF ↗ 17Replit and Other AI App BuildersPDF ↗ 18Figma and AI Design ToolsPDF ↗ 19Notion, PRD Tools and AI DocumentationPDF ↗ 20Automation with Zapier, Make and n8nPDF ↗ 21AI Chatbots for BusinessPDF ↗ 22Voice Agents for BusinessesPDF ↗ 23Learning Paths, Courses and CommunitiesPDF ↗

Business AI Stack (chapters 24 to 31)

24The business AI stack: tools and where to learn themPDF ↗ 25Next.js: customer-facing dashboardsPDF ↗ 26Zoho CRM: managing customers and leadsPDF ↗ 27WhatsApp Business Platform: customer messagingPDF ↗ 28Deployment with Vercel and AWSPDF ↗ 29Jira: tracking product and development workPDF ↗ 30Meta Business Suite, social media marketing and Google AdsPDF ↗ 31A one-page website that generates leadsPDF ↗

Complete AI tool guides (15)

1n8n: workflows, Gmail and an AI reply agentPDF ↗ 2Make: scenarios, routers and lead routingPDF ↗ 3Zapier: invoices from Gmail to Drive and a sheetPDF ↗ 4Webhooks: how apps talk in real timePDF ↗ 5Replit: one app for the web and a phonePDF ↗ 6Lovable: a job application trackerPDF ↗ 7Figma: components, auto layout and handoffPDF ↗ 8Claude Code: an AI coding agent in your projectPDF ↗ 9MCP: connecting AI assistants to your toolsPDF ↗ 10LLMs: how they work and how to call themPDF ↗ 11RAG: answers from your own documentsPDF ↗ 12AI agents: a job-search agent with approvalsPDF

↗ 13Agentic AI: planning, guardrails and teamsPDF ↗ 14Generative AI: uses, limits and a weekly reportPDF ↗ 15Other AI tools: choosing one per jobPDF ↗

Complete AI tool guides (one PDF, 30 pages) ↗

Libraries

★Framework Library*29 frameworks with worked examples and templates*PDF ↗ ★Case Study Library*12 product case studies*PDF ↗ ★Practical Projects and Portfolio*5 end-to-end portfolio projects*PDF ↗ ★Interview Handbook*100+ PM interview questions with model answers*PDF ↗ ★AI Tools Lab*Every AI tools class with step-by-step labs*PDF ↗

PM interview practice (free, on the original sites)

Hand-picked public question banks and guides from other PM communities. Each link opens the original page in a new tab.

PM frameworks cheat sheet

Framework	Use it to	How it works
RICE	Prioritise a backlog	Reach × Impact × Confidence ÷ Effort. Score each idea, then sanity-check the ranking.
Kano	Decide which features delight	Classify features as Must-be, Performance, Attractive, Indifferent or Reverse using a short survey.
MoSCoW	Scope a release	Must have, Should have, Could have, Won't have (this time).
Jobs-to-be-done	Understand motivation	When [situation], I want to [motivation], so I can [outcome].
AARRR (Pirate metrics)	Map the growth funnel	Acquisition, Activation, Retention, Referral, Revenue.
HEART	Measure user experience	Happiness, Engagement, Adoption, Retention, Task success, each with goals, signals and metrics.
North Star metric	Align the team	One metric that captures the value users get, supported by input metrics the team can move.
Five whys	Find the root cause	Ask 'why?' repeatedly until you reach a cause you can act on.
CIRCLES	Answer product design questions	Comprehend, Identify users, Report needs, Cut through prioritisation, List solutions, Evaluate trade-offs, Summarise.

PRD template

PRD: [Feature name] Owner: [Name] · Status: Draft · Last updated: [Date] 1. Problem Who has the problem, what it is, and the evidence (data, interviews, tickets). 2. Goal and success metrics Primary metric (with baseline and target), secondary metrics, guardrails. 3. Users and use cases Target segment, top 3 user stories: "As a [user], I want [action] so that [benefit]." 4. Solution What we'll build (link designs), what's out of scope, open questions. 5. AI behaviour (if applicable) Inputs, model, prompt/approach, what it must never do, human review, evaluation set and pass criteria, cost per request. 6. Launch plan Experiment design, rollout stages, support and comms, rollback criteria. 7. Risks and dependencies Technical, legal/privacy, operational; owners and mitigations.

Free courses