Marketeering.AI — AI Content Creation Playbook for AWS Partners (Free Listing)

A practical, model-agnostic framework for building a repeatable AI-assisted content engine across AWS Bedrock, Nova, SageMaker, and HubSpot/Zift.

Purpose & Scope

Equip AWS Partners (ISVs, SIs, and consulting practices) with a repeatable, compliant workflow to ideate, draft, review, and distribute GTM content using Generative AI.

This playbook is intentionally model-agnostic and focuses on safe defaults for India-based and global partners leveraging AWS services. It doubles as a resource attachment for an AWS Marketplace free listing.

Reference Architecture (Bedrock + Nova-first, with optional SageMaker)

Goal: Centralize content sources, enable Retrieval-Augmented Generation (RAG), enforce safety guardrails, and orchestrate asset creation to multi-channel delivery.

- Content lake: Amazon S3 for raw assets (PRDs, release notes, case studies, architecture diagrams, PDFs).
- Indexing: Amazon Bedrock Knowledge Bases or Amazon Kendra; for vector DB, consider Amazon OpenSearch Serverless (vector search) or Aurora PostgreSQL + pgvector.
- Core generation: Amazon Bedrock Converse API with Anthropic Claude, Meta Llama, or Mistral models. Prefer **Amazon Nova** family where available for multimodal/reasoning use cases.
- Safety & governance: Guardrails for Amazon Bedrock (blocklists, topic filters, PII redaction); Amazon Comprehend for extra PII/PHI detection on uploaded docs.
- Tool use & functions: Bedrock tool-calling; AWS Lambda for retrieval and orchestration; Step Functions for multi-step pipelines (draft → QA → approve → publish).
- Serving endpoints (optional): Amazon SageMaker for custom fine-tuned models or specialized inference (e.g., domain adaptation, latency control).
- APIs & events: Amazon API Gateway + Lambda for external triggers (HubSpot/Zift/Storylane form submissions); Amazon EventBridge for scheduled content refreshes.
- Observability & audit: Amazon CloudWatch logs + metrics; S3 versioning for content diffs; access via IAM roles with least privilege.

Model Selection Guide (Bedrock)

Pick a model by task, latency, cost, safety, and tool-use reliability. The table below offers defaults that are commonly available via Amazon Bedrock.

- Anthropic Claude (via Bedrock): strong long-context reasoning, safe defaults; great for structured docs, executive summaries, and tone adaptation.
- Meta Llama (via Bedrock): strong general-purpose writer, cost-effective; good for shorter assets and bulk generation.

- Mistral (via Bedrock): fast/lightweight; useful for high-volume paraphrasing and pattern-based transformations.
- Amazon Nova (where available): frontier multimodal/reasoning family; useful when mixing text + images/diagrams or requiring complex tool use.
- Amazon Titan (where applicable): Amazon's own models for embeddings, image, and some text tasks; great for retrieval pipelines.

OpenAI models are not provided natively on Bedrock; when required, integrate securely via the OpenAI API (or Azure OpenAI) behind VPC egress controls and secrets management. Articulate compliance rationale (data handling, no training on inputs, encryption in transit) in your architecture notes.

Methodologies AWS Reviewers Expect

Include explicit references to the methodologies and controls you apply:

- Retrieval■Augmented Generation (RAG): use Bedrock Knowledge Bases/Kendra and vector search to ground outputs in your own content.
- Guardrails & Safety: Guardrails for Bedrock, PII redaction, allow/deny lists; document your policy sets and test cases.
- Human-in-the-Loop (HITL): mandatory editorial checkpoints; human review for claims, metrics, and compliance statements.
- Structured Outputs: JSON Schema or XML tags for deterministic formatting; schema-validated responses (e.g., titles, bullets, callouts).
- Evaluation & QA: automatic linting and rubric scoring (helpfulness, accuracy, tone, brand safety) plus manual sign-off.
- Observability: prompt+response logging with secrets redaction; lineage from source doc → final asset.

Prompt Engineering — Reusable Templates

Adopt a 4-part prompt: Role • Task • Data • Output. Keep system instructions stable, pass source excerpts as context, require JSON outputs when helpful.

- **System Role (stable):** "You are a senior AWS GTM writer. Follow AWS Marketplace policies. Do not invent facts. Prefer grounded content."
- **Task (specific):** e.g., "Turn the release notes into a 600-word solution brief for CIOs."
- **Data (context):** paste excerpts, links, architecture bullets, KPIs. Mark as ….
- **Output Format:** require a schema: {title, audience, summary, bullets[], CTA}. Ask for valid JSON only.

Starter templates:

- • **PRD → Blog (Architect audience)**: Emphasize architecture, components, and SLAs; include a diagram caption and a 'How it works' section.
- • **Release Notes → Customer Update**: What's new, why it matters, rollout dates, impact on costs/performance, links to docs.

- • **Case Study Generator**: Problem → Approach → Architecture → AWS Services → Results (metrics) → Quote → CTA.
- • **Solution Brief (1■pager)**: Problem, Who it’s for, Architecture (bullets), Capabilities, Differentiators, CTA.

RAG Playbook (Nova/Claude on Bedrock)

Standard flow for grounded content:

- Ingest: push source docs to S3. Extract with AWS Lambda; normalize to text/markdown.
- Index: create Knowledge Base (or Kendra) with embeddings (Titan or Cohere/Mistral embeddings as supported).
- Retrieve: for each prompt, retrieve top■k chunks with metadata (source, date, owner).
- Generate: call Bedrock Converse API (e.g., Claude, Llama, or Nova) with retrieved context inserted into tags.
- Cite: require the model to return source_attributions[] with file and line hints.
- Validate: run a QA Lambda (regex + business rules) and a ‘factuality check’ prompt over the draft before human review.

Workflow Orchestration

- Draft: API Gateway/Lambda endpoint triggers generation when a PM uploads a doc or fills a HubSpot form.
- QA: Step Functions branch for (a) format validation, (b) safety filter, (c) rubric score.
- Review: Notify editor via SNS/Email with a signed S3 link to preview; capture edits in Git/Confluence.
- Publish: On approval, auto-push to CMS (S3/CloudFront, WordPress, HubSpot, Zift) via connectors.
- Versioning: Store prompts, model IDs, temperatures, and seeds to reproduce outputs later.

Evaluation & KPIs

- Speed: hours from input → approved asset (target: -60%).
- Accuracy: % assets passing factual QA first try (target: ≥90%).
- Reuse: % assets repurposed into ≥2 formats (target: ≥70%).
- Engagement: CTR or dwell vs. baseline (target: ≥2×).
- Cost per asset: <$ value benchmark per type (set internally).

Security, Privacy & Compliance Notes

- No training on customer content unless explicitly contracted; use inference-only policies.
- Encrypt in transit and at rest; apply VPC endpoints for Bedrock/Kendra/OpenSearch where possible.
- Mask or remove PII; store only minimum prompt/response logs required for audit, with secrets redaction.

- Document your content policy, escalation paths, and incident response for hallucinations or sensitive-topic failures.

MVP: 10-Day Stand-Up

- Day 1–2: S3 buckets, IAM roles, Guardrails for Bedrock, create Knowledge Base/Kendra.
- Day 3–4: Build Lambda retrieval+generation function; wire Bedrock Converse API (Claude/Llama/Nova).
- Day 5: JSON output schema + QA checks; CloudWatch logging; errors to DLQ.
- Day 6–7: Create 4 prompt templates (PRD→Blog, Release→Update, Case Study, Solution Brief).
- Day 8: Add editor review step (email/SNS), publish to S3/HubSpot/WordPress.
- Day 9–10: Backfill 3 existing assets; baseline metrics; cost dashboard.

Appendix — JSON Output Schema (Example)

Require models to return valid JSON for deterministic formatting:

```
{
  "title": "string",
  "audience": "e.g., CIO, VP Engineering, AWS Alliance Lead",
  "summary": "100-150 words",
  "sections": [
    {"heading": "string", "body_md": "markdown-compatible content"},
    {"heading": "string", "body_md": "..."}
  ],
  "bullets": ["string", "string"],
  "cta": {"text": "string", "url": "https://..."},
  "source_attributions": [{"doc": "s3://bucket/key", "lines": "L12-L45"}],
  "disclaimers": ["No billing through AWS Marketplace", "Outputs require human editorial review"]
}
```

Note on OpenAI Usage

OpenAI models are not natively part of Amazon Bedrock. If you integrate them, route traffic via a secure egress (NAT or VPC endpoint), store API keys in AWS Secrets Manager, and document data-handling posture (no data retention, encryption, SOC2/GDPR considerations).

Use the same JSON schema and QA controls so outputs stay interoperable with Bedrock workflows.