Hotel Assistant AI Agent — Key Settings Guide (Azure AI Foundry)

A practical, copy-paste-friendly checklist for configuring a production-ready hotel concierge/chat agent in Azure AI Foundry. Use this guide when creating a new Agent (or Prompt Flow–backed agent) and when promoting from Dev → Staging → Prod.

1) Project & Resource Setup

Goal: Clean separation of environments; secure connections to data; predictable deployments.

- Environments: Create separate resource groups or subscriptions for dev, staging, and prod.
- AI resources: Provision Azure OpenAI (models), Azure AI Search (vector index), Azure Storage (blobs), Azure Key Vault (secrets), Application Insights / Monitor (telemetry).
- Networking: Use private endpoints or VNET integration if PII is involved; configure IP allowlists and egress rules.
- RBAC: Apply least-privilege roles (Owner/Contributor for platform ops, Reader for stakeholders, custom roles for evaluators).
- Secrets: Store API keys and connection strings in Azure Key Vault and reference them through Foundry connections.

2) Agent Identity & Policy

Where: Agent → Basics / Instructions / Safety

- Agent Name: Hotel Assistant — JDM Hotels
- Short Description: Helps guests with reservations, check-in/out, amenities, and local recommendations.
- System Instructions (Persona): Tone is warm, professional, and hospitality-forward. Style is concise and actionable. Jurisdiction-aware policies (cancellation, taxes, fees). Privacy: never share room numbers or PII; verify identity before booking changes.
- Guardrails: Block payment collection; route payments to secure PCI-compliant flows. Disallow medical or legal advice. Refuse unsafe content politely and apply content filters.
- Language: Default English; auto-detect and respond in guest language (ES/FR/DE/PT). Confirm critical details in English for back-office handoff.

Template — System Prompt (paste into Instructions)

You are JDM Hotels' virtual concierge. Prioritize guest safety, privacy, and accuracy.
- Always verify identity for booking lookups or changes (last name + confirmation code or phone).
- Never disclose room numbers or personal data. Never accept payments; route to secure checkout URLs.
- Be proactive but concise. Offer next-best actions and summarize confirmations.

- For operational requests (extra towels, late checkout), create a service ticket with priority and ETA and confirm back to the guest.
- Detect language and reply in that language. For critical confirmations, include an English summary line prefixed with 'Back-office:'.
- If a request is unsafe or out of policy, refuse politely and suggest a safe alternative.

3) Model & Inference Settings

Where: Agent → Model / Parameters

- Chat model: GPT-5-mini (or latest equivalent). Maintain both fast and quality variants if available.
- Temperature: 0.3–0.5; Top-p: ~0.9 for balanced consistency.
- Max output tokens: 512–800 for chat; 1,200+ for summaries or itineraries.
- Response format: Plain text for conversation; JSON schema for function/tool outputs.
- Streaming: Enabled for responsiveness.

4) Knowledge & Retrieval (RAG)

Where: Agent → Add Data / Grounding (Azure AI Search, Blob Storage)

- Sources: SOPs, amenities, room types, fees, house rules, loyalty tiers, menus, local attractions, emergency procedures.
- Indexing: Chunk size 500–1,000 tokens with 60–120 token overlap. Use latest text embedding model.
- Fields: title, content, language, effective_date, property_code, policy_version.
- Validation: After updates, run sample guest queries to confirm the latest policy is referenced.

5) Testing, Evaluation & Monitoring

- Playground: Rapidly test tone, grounding, and edge cases before deployment.
- Evaluation: Use standard queries to compare prompt or personality variants (A/B testing).
- Edge cases: Test emergencies, ambiguous requests, multilingual queries, and low-confidence scenarios.
- Monitoring: Track repeated questions, confidence scores, latency, and fallback usage.

6) Deployment & Security

- Deployments: Create versioned deployments and route traffic via endpoints.
- Authentication: Use API keys or OAuth; rotate secrets regularly.
- Encryption: Encrypt data in transit and at rest.
- Error handling: Implement retries, graceful degradation, and human escalation for high-risk cases.