Microsoft

Business-Technical Alignment Checklist for Microsoft Foundry

To build effective AI agents, it's critical to achieve business-technical alignment. The checklist below provides actionable steps to keep cross-functional teams aligned on goals, using Foundry as the central platform.

This list is organized by the five pillars of the Microsoft Well-Architected Framework:

Security | Reliability | Performance Efficiency | Cost Optimization | Operational Excellence

Security

Embed governance and compliance early	Implement Azure Policy governance to regulate model deployments. Implement the Azure Security Baseline. Monitor the baseline and its recommendations with Microsoft Defender for Cloud.
Implement unified identity and access control	Integrate Microsoft Entra ID multi-tenant authentication for all agent portals and APIs. Set up a Microsoft Entra ID cloud-based identity and access management service to provide the essential identity, authentication, policy, and protection to secure the offer. Apply consistent Role-Based Access Control (RBAC) so business and IT have proper visibility and access rights.
Secure integration touchpoints	Design the agent to fit securely into existing systems. Plan API keys, webhooks, and data pipelines to use Azure Key Vault and TLS encryption. Involve security teams to validate the agent's data flows and ensure it meets corporate network and zero-trust requirements before full development. Scan your model and application endpoints with AI Red Teaming Agent. You can automatically scan your model, evaluate probing success, and generate a score card to understand your level of deployment readiness.
Document data usage and privacy	Document the agent's data footprint and protection plan, and align logging and storage decisions with business risk tolerance and applicable standards.

Reliability

Define clear objectives and success KPIs	Use Microsoft-recommended agent evaluators and general-purpose evaluators as a guide. Create a written agreement that maps business performance goals to requirements for building and running these evaluators.
Plan for disaster recovery and resiliency	Plan a multi-region deployment of Foundry and associated resources. Maximize your chances to recover logs, notebooks, Docker images, and other metadata. Design your solution for high availability. Fail over to another region. Prioritize implementing Disaster Recovery for Hub Projects. Use Foundry's project workspace to demo agent behavior against real use cases. Frequent feedback cycles catch misalignment and reliability issues early, before they become expensive rework.
Plan for end-to-end integration testing	Plan for integration tests that mirror real business workflows (like connecting the agent to sample CRM or ERP data) in a staging environment. This ensures the agent works reliably within the company's ecosystem and avoids "works in isolation" failures when you go live.
Implement user feedback loops	Run a pilot with a small group of end users/stakeholders on interim builds; fold their qualitative feedback into each sprint to validate that the agent's outputs deliver business value—not just technical accuracy.

Performance Efficiency

Design for actual scale needs	Scale for realistic early load (e.g., if you have ~50 users, do not build for 50,000 users on day one) and use Foundry's evaluation tools to select model sizes and endpoints that balance performance and capacity.
Prioritize core features first	Prioritize features that align with defined business goals to reduce unnecessary complexity. Use Azure Boards to collaborate and follow up with Foundry project management.
Optimize model and infrastructure choices	Select AI models and Azure services that balance performance and cost. For instance, use a smaller, faster model if it meets the accuracy requirements. Foundry's integrated Leaderboards let the team evaluate latency/throughput of different approaches together, aligning technical trade-offs with business SLA expectations.
Avoid premature optimizations	Start simple, measure performance, and optimize based on telemetry in Foundry dashboards. Defer advanced load-balancing until sustained demand is observed.

Cost Optimization

Set joint budget guardrails	Before development, agree on budget constraints such as Azure usage and development hours. Configure Foundry's cost management alerts or quotas to enforce these limits. This ensures technical teams remain aware of cost as a factor in design decisions and stay on track to meet business ROI targets.
Implement real-time cost monitoring	Use Foundry's cost dashboards to track cloud spend as features ship and share a weekly cost snapshot in team meetings (e.g., "enabling Feature-X increased compute by ~20%"). This transparency allows the team to weigh business value versus cost and decide to keep, tune, or roll back the feature.
Tie features to ROI	Build only what pays its way: Prioritize features with clear business value and ensure development costs align. If a feature is high-cost and low-impact, consider adjusting scope early to maintain cost efficiency.
Leverage Azure credits and MACC	If Azure credits are available or your solution qualifies for Microsoft Azure Consumption Commitments (MACC), consider incorporating these resources into your planning. For instance, credits can help offset expenses such as initial training or data processing. Leveraging these options may help manage costs and support development timelines; actual savings and outcomes will vary based on usage and configuration.

Operational Excellence

Establish cross-functional cadence	Set up a joint working model: Include business and technical owners in sprint planning, backlog grooming, and retrospectives. Use Foundry's project workspace as a **single hub** where both groups can see progress and issues. A well-defined cadence prevents silos and keeps everyone in sync on priorities and next steps.
Create a shared project workspace	Single source of truth: Maintain a single source of truth by documenting business and technical requirements together in Foundry's wiki or OneNote, and logging changes in strategy or scope for stakeholder visibility.
Integrate governance and DevOps	Incorporate security reviews, quality gates, and compliance checks into the CI/CD pipeline with GitHub Actions or Azure DevOps. For example, require sign-off from both a business lead and a technical lead before deploying to production. This integrated approach ensures operational processes serve both business quality standards and technical reliability.
Practice iterative delivery and improvement	Adopt an agile approach that aims to deliver a minimum viable product (MVP) first. Deliver a usable agent quickly, then enhance it in iterations based on real-user data and feedback. Continuously track outcomes using Foundry's analytics and custom business KPIs, and refine the backlog based on those insights. This approach may accelerate time-to-market as well as keeping the product aligned with evolving business goals.
Promote stakeholder visibility and buy-in	Share regular, concise updates to all stakeholders to support leadership + engagement, enable timely decisions, and make the business alignment of technical work visible.

Learn more about building and monetizing AI agent solutions with the Microsoft AI Envisioning Day video series.

Watch now >

Now that your teams are aligned, you can draft, configure, and publish your app to the Microsoft Marketplace.

Learn how >