Accelerating software development lifecycles on AWS with generative AI

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Accelerating software development lifecycles on AWS with generative AI

Table of Contents

Accelerating software development lifecycles on AWS with generative AI

Chetan Makvana, Amazon Web Services

April 2025 (document history)

Growing demand for high-quality software is driving organizations to constantly seek ways to accelerate their software development lifecycle (SDLC). As organizations strive to remain competitive, it is critical to reduce the time to market while maintaining or improving product quality. To meet these challenges, the software development experience must evolve and use cutting-edge technologies, methodologies, and practices that streamline processes and empower software development teams to be more productive and creative. The emergence of the next generation development experience marks a significant shift in how software is conceived, built, tested, and deployed. It integrates a variety of capabilities—including cloud-native development, AI-driven automation, advanced project management, collaborative tools, and DevSecOps—that collectively enhance the efficiency and effectiveness of the SDLC.

At the forefront of this transformation is the rise of generative AI in software engineering. According to Gartner, 40% of platform engineering teams will use AI to augment every phase of the SDLC by 2027, compared to only 5% in 2023. This report also states that software engineering leaders must now prepare to adopt generative AI across a broader range of areas that are critical to the development process. In another report, McKinsey research shows that companies with a higher developer velocity index grow revenue 4–5 times faster, have 60% higher shareholder returns, and are 55% more innovative. By embracing generative AI beyond just code generation, organizations can unlock new level of efficiency, productivity, and innovation in their software development workflows. This can reduce manual effort, surface insights, and augment human expertise.

Objectives

This strategy document outlines a framework, foundational capabilities, use cases, best practices, and success metrics that can help you accelerate your SDLC with generative AI. It describes how to effectively integrate generative AI across all development stages in order to improve product quality and efficiency.

This strategy document can help you and your organization achieve the following objectives:

- Implement a framework, foundational capabilities, use cases, best practices, and success metrics to accelerate your SDLC with generative AI.
- Effectively integrate generative AI across all development stages to improve product quality, release velocity, and development efficiency.
- Adapt to the next generation of software development by incorporating cutting-edge AI technologies, methodologies, and practices that streamline processes and empower development teams.

Intended audience

This strategy document is for IT leaders, engineering managers, chief technology officers, and software development teams who want to accelerate their software development lifecycle by applying generative AI to their development practices.

Understanding the software development experience

The *software development experience* encompasses the environment, tools, and processes your development teams use throughout the software development lifecycle (SDLC). It includes the integrated development environment (IDE), collaboration platforms, testing frameworks, knowledge management systems, deployment pipelines, and more.

A well-designed development experience streamlines workflows, reduces manual effort, and empowers your teams to focus on high-value tasks, which ultimately accelerates your SDLC. For example, by seamlessly integrating your IDE, version control system, and deployment tools, you enable developers to write, test, and deploy code with greater speed and efficiency compared to a fragmented toolchain that requires manual handoffs and context switching. Similarly, integrating a robust knowledge management framework helps teams easily access and share institutional knowledge, best practices, and documentation. This enhances their overall productivity and problem-solving capabilities.

The software development experience has a direct impact on the overall performance and success of a software development team. A sub-optimal experience can lead to the following:

- **Reduced productivity** – Inefficient tools, complex workflows, and lack of automation hamper team productivity, which slows the delivery of features and updates.
- **Increased technical debt** – Poorly integrated tools and ad-hoc processes can result in technical debt, which makes it more challenging to maintain and scale your software systems over time.
- **Diminished innovation** – When bogged down by manual, repetitive tasks, your team's ability to explore new technologies and drive innovation is constrained.
- **Compromised quality** – Fragmented testing and deployment processes increase the risk of software defects and vulnerabilities. This can negatively affect the overall quality of the delivered software.

By investing in a well-designed software development experience, you can unlock significant benefits, such as faster time to market, improved software quality, enhanced software development team satisfaction, and greater business agility.

Powering the software development experience with generative AI

The integration of generative AI into the software development lifecycle (SDLC) represents a paradigm shift in how entire software development teams conceive, design, implement, and maintain software solutions. Generative AI has the potential to revolutionize every phase of the SDLC, including project management, requirements gathering, design, coding, testing, deployment, and maintenance.

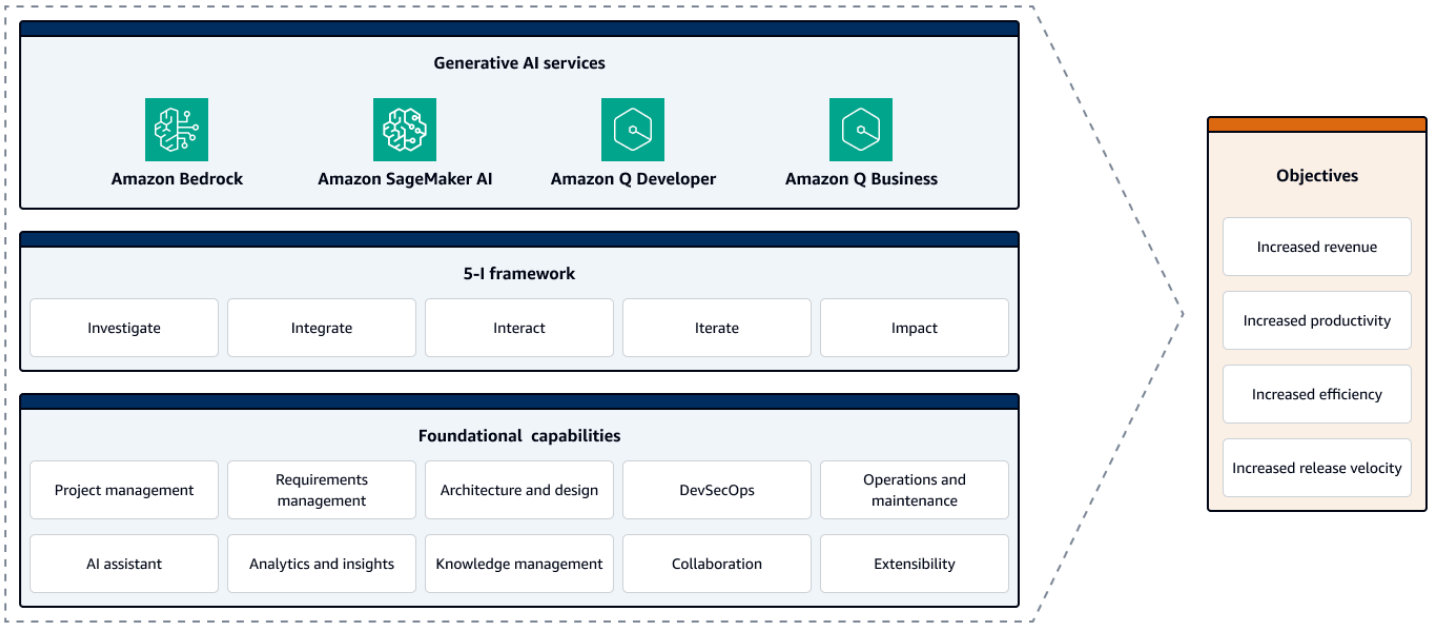
At its core, a generative AI-powered development experience acts as an intelligent collaborator for your entire software development team, including product managers, designers, solutions architects, developers, testers, and operations personnel. It provides context-aware assistance, generates artifacts (such as user stories, design mock-ups, code snippets, and test cases), offers near real-time suggestions, and even predicts potential issues before they arise. This AI-augmented approach significantly reduces the cognitive load on team members. This allows them to focus on high-level strategic decisions and complex problem-solving while generative AI handles the more mundane, repetitive tasks.

Generative AI also serves as a knowledge amplifier. It helps team members quickly access relevant information, best practices, and patterns from vast repositories of data. This can effectively democratize expertise across the organization. By seamlessly integrating generative AI capabilities throughout the development toolchain, you can create a more intuitive, efficient, and productive environment for your entire software development teams. This enhanced development experience accelerates the SDLC and improves overall quality. It also reduces errors and fosters innovation because team members can explore new ideas and approaches more rapidly.

To adopt a generative AI-powered development experience in your organization, consider the following key elements:

- 5-I framework – Consisting of five dimensions, the 5-I framework provides a comprehensive approach to navigate the process of modern software development. It offers a structured methodology that helps you systematically apply generative AI across all stages of the SDLC.
- Foundational capabilities – To fully use the power of generative AI across the dimensions of modern software development, you need to establish a robust set of foundational capabilities. These capabilities form the backbone of an AI-powered development experience. These capabilities help you integrate and use generative AI throughout the SDLC.

Together, the 5-I framework and foundational capabilities form a strategy for reimagining the software development experience. The five dimensions provide a strategic framework for applying generative AI, and the foundational capabilities prepare your organization to support this AI-driven approach. AWS services, such as Amazon Bedrock, Amazon SageMaker AI, Amazon Q Developer, and Amazon Q Business, provide generative AI capabilities and features that you can integrate into your software development experience.



5-I framework for an AI-powered software development experience

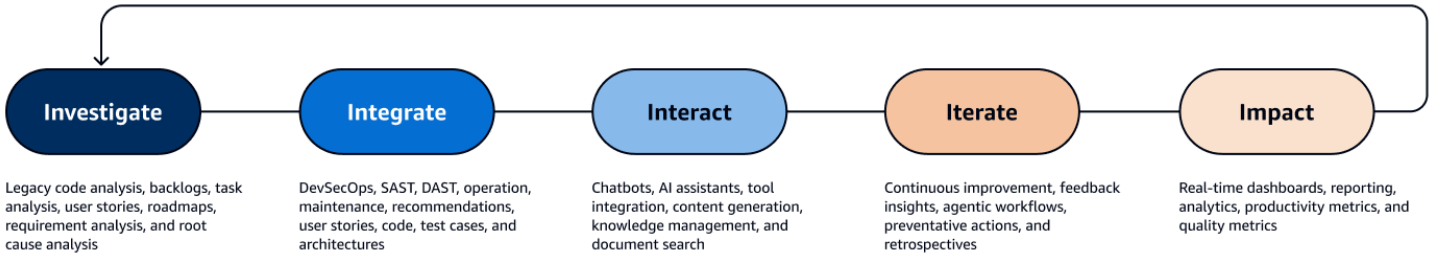
The 5-I framework provides a structured approach for software development teams to effectively integrate generative AI into their development practices. It helps you establish a robust foundation for using generative AI throughout the SDLC. It also helps you set up the right development practices, workflows, and mindsets to fully harness the potential of generative AI.

This section contains the following topics:

Framework overview

The 5-I framework is built around five key dimensions: Investigate, Integrate, Interact, Iterate, and Impact. Each dimension represents a critical area where generative AI significantly enhances the software development process. By strategically integrating generative AI across these dimensions, the framework addresses the evolving needs of modern software development. It can reduce cognitive load and amplify creative potential. It recognizes that the ideal development experience is not just about tools—it's about creating an environment where AI seamlessly enhances human capabilities at every stage.

The following diagram shows the five dimensions of AI-powered software development. For each dimension, it shows where you can integrate generative AI in order to drive efficiency and innovation.



The following are the five dimensions in the framework:

- **Investigate** – Enhance every analytical task in your software development process with generative AI. Use generative AI to understand requirements, process vast amounts of data, recognize patterns, and generate insights that might be beyond human capacity or would take significantly longer to produce. These insights help you make more informed decisions, quickly identify improvement opportunities, and more efficiently deliver high-quality software. Generative AI can be an intelligent partner for the analytical processes throughout the SDLC. By harnessing generative AI, you apply in-depth analysis to critical areas, such as requirements gathering, legacy codebase examination, and product backlog optimization. For example, product owners can use generative AI to analyze user journeys or requirements before creating user stories. Development teams can uncover inefficiencies and identify optimization opportunities in existing codebases. DevOps engineers can apply root cause analysis to quickly diagnose performance issues or security vulnerabilities, which can improve reliability.
- **Integrate** – Integrate generative AI to automate a wide range of tasks and processes across the entire SDLC. This includes automatically generating code snippets, test cases, architectural designs, user stories, and deployment pipelines. By automating these typically manual tasks,

teams can focus on more strategic and innovative work, which drives faster time to market and high-quality applications. The Integrate dimension represents a paradigm shift in software development, where AI becomes an integral part of the development process. It works alongside your software development team to enhance productivity, improve quality, and drive innovation. This results in faster time to market. It challenges your software development teams to regularly assess their processes and workflows by asking at each step: "Can this be automated?"

- **Interact** – Use generative AI-powered assistants to provide your team with instant, contextual support across a range of tasks and queries. These intelligent assistants act as knowledgeable collaborators that draw from a vast repository of information. They can answer coding questions, offer design suggestions, explain standard operating procedures, and help troubleshoot complex issues. Integrating these AI assistants into the development workflow boosts productivity and fosters a more collaborative, problem-solving environment.
- **Iterate** – Use generative AI to enable rapid, data-driven adjustments throughout the SDLC. You can continuously analyze data from sources such as customer feedback, usage patterns, market trends, and team performance metrics in order to make informed decisions quickly. This adaptability refines your software development from a static, predefined process into a fluid, responsive approach. It manifests in various ways, including dynamic prioritization of backlogs, flexible resource allocation, adaptive testing strategies, evolving documentation, and responsive deployment processes. For example, product managers can use AI-generated insights to reorder their backlogs, integrating new customer requirements and market trends in near real time. DevOps engineers can adapt deployment plans and infrastructure configurations based on performance analytics, making sure that applications remain resilient and optimized. Development teams can translate feedback from sprint retrospectives into actionable improvements for the next iteration, driving a culture of continuous process enhancement.
- **Impact** – Apply generative AI to assess the effectiveness and performance of your software development process. By using AI-powered analytics and metrics, you gain deeper insights into development efficiency, code quality, user engagement, and overall application performance. This data-driven approach helps you make informed decisions, optimize your development workflows, and continuously improve the quality and user experience of your applications. When assessing software team productivity, generative AI analyzes various data points, such as code commit frequency, issue resolution times, release velocity, feature delivery rates, and more. It can also evaluate the quality of code reviews, the effectiveness of collaboration tools, and the impact of different development practices on the overall team output. By correlating these metrics with project outcomes, the AI identifies patterns and trends that human analysts might miss, and they can provide actionable insights that boost team productivity. Furthermore, generative AI can

help you benchmark team performance against industry standards or historical data, offering personalized recommendations for improvement. It can also predict potential bottlenecks or risks in the development process so that you can take proactive measures.

Integrating with the software development lifecycle

The SDLC consists of multiple phases, which can differ from organization to organization. Commonly, these phases include the following: requirements and planning, design and architecture, implementation, testing, deployment, and operation and maintenance.

The following table maps the dimensions of the 5-I framework to the SDLC phases and provides the level of integration for each dimension.

Framework dimension	Requirements and planning	Design and architecture	Implementation	Testing	Deployment	Operation and maintenance
Investigate	High	Low	Low	Low	Low	Medium
Integrate	Medium	Medium	High	Medium	High	High
Interact	High	High	High	Medium	Medium	High
Iterate	Medium	Low	Low	Low	Low	Medium
Impact	High	Medium	High	Low	High	High

The levels of integration vary from high to low. The mapping reveals key focus areas for each dimension. For instance, *Investigate* shows high intensity in the requirements and planning phase. *Integrate* demonstrates high intensity in the implementation, deployment, and operation and maintenance phases.

By using this mapping, you can prioritize your efforts effectively. We recommend that you focus on high, then medium, and then low. Make sure that you adopt a balanced and impactful approach that enhances the software development experience with generative AI.

Foundational capabilities for an AI-powered software development experience

To successfully implement a generative AI-powered software development experience, you need to establish a set of foundational capabilities that span multiple personas in your organization. These capabilities represent your ability to effectively deploy resources, implement processes, and achieve desired outcomes in the context of AI-powered software development. By cultivating these capabilities, you create a robust foundation that helps you seamlessly integrate generative AI across all stages of the SDLC.

AWS provides key services to help you implement these capabilities. For example, Amazon Q Developer helps accelerate software development by acting as an AI-powered assistant. Amazon Q Business helps you get fast, relevant answers to pressing questions, solve problems, and generate content. It can also act on your behalf by integrating tools related to software development. Amazon Bedrock provides access to foundation models and broad set of capabilities to customize specific development workflows and requirements.

By cultivating these capabilities through AWS services, you create a robust foundation that helps you seamlessly integrate generative AI across all stages of the SDLC.

The following are the foundational capabilities that you should focus on:

- Project management
- Requirement management
- Architecture and design
- Collaboration
- DevSecOps
- Operation and maintenance
- AI assistants
- Analytics and insights
- Knowledge management
- Extensibility

Each foundational capability integrates with the framework dimensions and the different stages of the SDLC. This integration helps you use AI capabilities effectively throughout your

software development process. It enhances efficiency, quality, and innovation at every step. The synergy between these foundational capabilities, the framework, and the SDLC stages creates a comprehensive ecosystem for AI-powered software development. This helps you harness the full potential of generative AI, drive continuous improvement, accelerate development cycles, and deliver quality software products.

The following table shows how the foundational capabilities and subcapabilities map to the framework dimensions and the SDLC phases.

Capability: subcapability	**Investigate**	**Integrate**	**Interact**	**Iterate**	**Impact**
Project management: Issue management	Requirements and planning	None	None	None	None
Project management: Sprint and task management	Requirements and planning	Requirements and planning	None	None	None
Project management: Product backlog management	Requirements and planning	None	None	Requirements and planning	None
Project management: User stories mapping	Requirements and planning	None	None	None	None

Capability: subcapability	Investigate	Integrate	Interact	Iterate	Impact
Project management: Reporting and analytics	Requirements and planning	None	None	None	Requirements and planning
Project management: Product roadmap management	Requirements and planning	None	Requirements and planning	None	None
Project management: Feedback loops	None	None	None	Requirements and planning	None
Project management: Retrospectives	None	None	None	Requirements and planning	None
Requirement management	Requirements and planning	Requirements and planning	None	None	None
Architecture and design: Solution design	Design and architecture	Design and architecture	None	None	None

Capability: subcapability	Investigate	Integrate	Interact	Iterate	Impact
Collaboration: Documentation management	All SDLC phases	None	All SDLC phases	None	None
Collaboration: Knowledge sharing	All SDLC phases	None	All SDLC phases	None	None
Collaboration: Project asset management	None	All SDLC phases	All SDLC phases	None	None
DevSecOps: CI/CD	Testing, Deployment	Implementation, Testing, Deployment	Deployment	None	None
DevSecOps: DevOps security	Implementation	Implementation, Testing, Operation and maintenance	None	Implementation, Testing, Operation and maintenance	None
DevSecOps: Application performance monitoring	None	Operation and maintenance	None	None	None

Capability: subcapability	Investigate	Integrate	Interact	Iterate	Impact
DevSecOps: Log aggregation and analytics	Operation and maintenance	Operation and maintenance	None	None	None
DevSecOps: AIOps	Operation and maintenance	None	None	Operation and maintenance	None
DevSecOps: Continuous improvement	None	None	None	Operation and maintenance	None
DevSecOps: Dashboard monitoring	None	Operation and maintenance	None	None	None
DevSecOps: Performance insights	Operation and maintenance	None	None	Operation and maintenance	None
Operation and maintenance: Incident management	None	None	None	Operation and maintenance	None
Operation and maintenance: Code upgrades	None	Operation and maintenance	None	None	None

Capability: subcapability	Investigate	Integrate	Interact	Iterate	Impact
Operation and maintenance: Code optimization	Operation and maintenance	Operation and maintenance	None	None	None
Operation and maintenance: Technical debt management	None	Operation and maintenance	Operation and maintenance	None	None
Operation and maintenance: Change management	None	Implementation, Deployment	None	None	None
Operation and maintenance: Reverse engineering	Operation and maintenance	None	None	None	None
Operation and maintenance: Code modernization	None	Implementation	None	None	None

Capability: subcapability	Investigate	Integrate	Interact	Iterate	Impact
Operation and maintenance: Performance optimization	None	Operation and maintenance	None	Operation and maintenance	None
Analytics and insights	None	Requirements and planning	None	None	All SDLC phases
AI assistant	None	None	All SDLC phases	None	None
Knowledge management	None	None	All SDLC phases	None	None
Extensibility	None	Deployment	None	None	None

Generative AI use cases for project management

Effective project management is at the heart of successful software development. In the context of generative AI, project management takes on new dimensions. It can become more predictive, adaptive, and data-driven. AI-powered project management tools analyze historical project data to generate more accurate time and resource estimates. They can automatically prioritize tasks based on business objectives and team capacity, and they can even predict potential roadblocks before they occur. For instance, a project manager might use generative AI to create a preliminary project plan based on the project's requirements and historical data from similar projects. The AI could then suggest optimal team compositions that account for skills, workloads, and project needs. Throughout the project, AI-driven dashboards provide near real-time insights into the project status by automatically generating reports and highlighting areas that require attention.

This AI-augmented approach to project management can enhance efficiency. It helps project managers focus on strategic decision making and team leadership, rather than getting bogged down in routine administrative tasks.

The following table shows project management use cases that you can enhance with generative AI and the persona responsible for those use cases.

Subcapability: Use case	Persona
Issue management: Create and assign issues	Project manager
Issue management: Detect issues during testing and log them	Test engineer
Issue management: Prioritize issues based on severity and assign them to developers	Project manager
Issue management: Identify and merge duplicate issues	Project manager
Issue management: Track and generate reports about key issues, metrics, and overall health of the project	Project manager
Sprint and task management: Estimate effort for tasks and assign story points based on team capacity	Scrum Master
Sprint and task management: Distribut e tasks among team members for even workload across the sprint	Scrum Master
Sprint and task management: Facilitate sprint planning sessions that align team efforts with sprint goals	Scrum Master
Product backlog management: Reorder backlog items based on business value, urgency, and user feedback	Product owner

Subcapability: Use case	Persona
Product backlog management: Integrate new customer feedback and market insights into the product backlog for near real-time prioritization	Product owner
Product backlog management: Identify and manage dependencies between backlog items to streamline development	Product manager
User stories mapping: Create maps of user journeys to identify all necessary features and their corresponding user stories	Product owner
User stories mapping: Identify gaps or missing steps in the user flow	Business analyst
User stories mapping: Prioritize user stories based on their impact to the business value	Product manager
Reporting and analytics: Generate near real-time dashboards that visualize key project metrics, such as sprint velocity and issue resolution rates	Project manager
Reporting and analytics: Analyze historical data and predict future project outcomes, such as potential delays or bottlenecks	Project manager
Reporting and analytics: Create custom reports, such as team performance or project status reports, that are tailored to different stakeholders	Project manager
Product roadmap management: Create and maintain a product roadmap that outlines major milestones and release dates	Project manager

Subcapability: Use case	Persona
Product roadmap management: Update the roadmap based on changes in project prioritie s or timelines	Product manager
Product roadmap management: Share the roadmap with stakeholders to provide visibilit y into the product's direction	Product manager
Feedback loops: Collect feedback from the team after each sprint and identify areas for improvement	Scrum Master
Retrospectives: Translate feedback into actionable items for the next sprint, driving continuous improvement	Scrum Master
Retrospectives: Track the impact of changes implemented from previous retrospectives to measure their effectiveness	Scrum Master

Generative AI use cases for requirement management

Requirement management is a critical process that is closely tied to project management. Imagine a product owner using an AI tool to analyze customer feedback, market trends, and stakeholder inputs. The AI tool could generate a comprehensive set of user stories and requirements, automatically categorize them, detect potential conflicts or gaps, and even suggest prioritization based on business value and implementation complexity. As the project progresses and requirements evolve, the AI can continuously update and refine the requirements to make sure that they remain aligned with changing business needs and technical constraints. This dynamic, AI-driven approach to requirement management helps make sure that development efforts remain tightly aligned with user needs and business goals throughout the project lifecycle.

The following table shows requirement management use cases that you can enhance with generative AI and the persona responsible for those use cases.

Use case	Persona
Create business requirements	Business analyst
Create epics from features	Product owner
Track the progress of an epic by monitoring the completion of its associated user stories	Product manager
Create user stories	Product owner
Estimate the effort required for each use story and assign story points	Scrum Master
Define acceptance criteria for each user story	Product owner

Generative AI use cases for architecture and design

With a solid foundation of project management and well-defined requirements, the next critical capability is architecture and design. Here, generative AI is opening up new possibilities for creating robust, scalable, and efficient software architectures. AI-powered design tools can analyze requirements and constraints to suggest optimal architectural patterns and design approaches. They generate multiple design alternatives, and each is optimized for different priorities, such as performance, scalability, or maintainability. For example, a solutions architect might use an AI assistant to quickly generate several high-level architectural designs based on the project requirements. This AI-augmented approach accelerates the design process and helps architects make more informed decisions. This leads to more robust and future-proof software designs.

The following table shows architecture and design use cases that you can enhance with generative AI and the persona responsible for those use cases.

Use case	Persona
Create an architecture document	Solutions architect
Create a detailed design document	Technical lead

Use case	Persona
Understand an existing architecture and design standards	Solutions architect
Develop detailed mock-ups and prototypes of a user interface	UX/UI designer

Generative AI use cases for collaboration

Software development is inherently a collaborative endeavor. You can use generative AI to enhance collaboration on your software development team. AI-powered collaboration tools go beyond simple messaging and file sharing. They facilitate more effective communication by summarizing long discussion threads, highlighting key decisions, and even suggesting optimal times for meetings based on team members' schedules and productivity patterns. AI can assist in code reviews by automatically identifying potential issues, suggesting improvements, and even explaining complex changes to reviewers. During brainstorming sessions, AI can act as a facilitator, generate ideas, help organize thoughts, and even mediate discussions to make sure that all voices are heard. For distributed teams, AI can help bridge cultural and language barriers. It can provide near real-time language translation in chat and video calls and offer cultural context to help prevent misunderstandings. By augmenting human collaboration with AI, this capability helps teams work more efficiently and effectively, which fosters innovation and improves overall project outcomes.

The following table shows how you can use generative AI to enhance collaboration use cases.

Subcapability: Use case	Persona
Document management: Create and maintain a centralized documentation repository	Technical writer
Document management: Allow multiple team members to collaborate on documentation in real time	Development team
Knowledge sharing: Use discussion forums as a platform for developers to ask questions	Development team

Subcapability: Use case	Persona
, share knowledge, and troubleshoot issues collaboratively	
Knowledge sharing: Use discussion forums to document and track decisions made during project discussions, making sure that the rationale behind key decisions is captured and accessible for future reference	Product manager
Project asset management: Facilitate easy sharing of project-related resources	Development team
Project asset management: Implement version control for shared content so that team members can track changes, revert to previous versions, and collaborate on content updates	Development team

Generative AI use cases for DevSecOps

AI-powered DevSecOps tools automate many aspects of the software delivery pipeline. For example, they can perform intelligent code reviews, detect potential bugs, detect security vulnerabilities, and identify performance issues in near real time as developers write code. AI generates and runs comprehensive test suites, and it automatically updates them as the codebase evolves. This AI-augmented approach to DevSecOps accelerates the delivery pipeline and significantly enhances the security and reliability of the software being delivered.

The following table shows DevSecOps use cases that you can enhance with generative AI and the persona responsible for those use cases.

Subcapability: Use case	Persona
DevOps and continuous delivery: Automated entire deployment pipelines	DevOps engineer

Subcapability: Use case	Persona
DevOps and continuous delivery: Receive near real-time feedback on code quality and potential issues	Software developer
DevOps and continuous delivery: Receive near real-time security issues and remediation recommendations	Software developer
DevOps and continuous delivery: Receive near real-time code and best practice suggestions	Software developer
DevOps and continuous delivery: Automate repetitive tasks and integrate commands into scripts	DevOps engineer
DevOps and continuous delivery: Build code and generate artifacts automatically after each code commit	Software developer
DevOps and continuous delivery: Build code according to the organization's standards and framework	Software developer
DevOps and continuous delivery: Automatic ally run unit tests on every commit to catch errors early in the development process	Software developer
DevOps and continuous delivery: Analyze the coverage of unit tests to make sure that all critical code paths are tested	Software developer
DevOps and continuous delivery: Manage branches and merge changes	Software developer

Subcapability: Use case	Persona
DevOps and continuous delivery: Manage code and artifact versioning	Software developer
DevOps and continuous delivery: Store and manage build artifacts and dependencies	DevOps engineer
DevOps and continuous delivery: Resolve and fetch dependencies during the build process	Software developer
DevOps and continuous delivery: Generate and run integration tests to make sure that components work together as expected	Test engineer
DevOps and continuous delivery: Use mock services during integration tests to simulate interactions with external systems	Test engineer
DevOps and continuous delivery: Benchmark application performance under different loads	Performance engineer
DevOps and continuous delivery: Simulate high-traffic scenarios to test the application's scalability and response times	Performance engineer
DevOps and continuous delivery: Test the system's ability to recover from failures, such as server crashes or network outages	Site reliability engineer
DevOps and continuous delivery: Perform chaos engineering	Site reliability engineer
DevOps and continuous delivery: Run tests to verify that the application meets the business requirements	QA engineer
DevOps and continuous delivery: Conduct user acceptance testing	Product owner

Subcapability: Use case	Persona
DevOps and continuous delivery: Scan dependencies for vulnerabilities and license compliance issues	Security engineer
DevOps and continuous delivery: Monitor and manage open source dependencies to make sure that they are up to date and secure	Security engineer
DevOps and continuous delivery: Generate and maintain a software bill of materials (SBOM) to track all components and dependencies	Security engineer
DevOps and continuous delivery: Use the SBOM to conduct audits for regulatory compliance	Compliance officer
DevOps and continuous delivery: Create release notes	Release manager
DevOps and continuous delivery: Plan and coordinate releases	Release manager
DevOps and continuous delivery: Implement standard operating procedures for rollback and release management	Release manager
DevOps and continuous delivery: Use feature flags to enable or disable features in productio n without deploying new code	Product manager
DevOps and continuous delivery: Run A/B tests using feature flags to measure the impact of different features on user behavior	Product manager

Subcapability: Use case	Persona
DevOps and continuous delivery: Analyze and monitor pipeline failures	DevOps engineer
DevOps and continuous delivery: Create and manage infrastructure resources	DevOps engineer
DevOps and security: Scan code repositories for hardcoded secrets	DevOps engineer
DevOps and security: Implement near real-time detection to alert developers immediate ly if secrets are committed to the repository	DevOps engineer
DevOps and security: Enforce continuous code quality monitoring	Software developer
DevOps and security: Detect and flag indicators of potential security vulnerabilities in code	Software developer
DevOps and security: Implement automated testing for Open Worldwide Application Security Project (OWASP) top 10 security risks to make sure that the application adheres to industry-standard security practices	Security engineer
DevOps and security: Regularly update and educate developers about OWASP risks by integrating checks into the development process	Security engineer
DevOps and security: Scan third-party libraries and dependencies for known security vulnerabilities	DevOps engineer

Subcapability: Use case	Persona
DevOps and security: Scan application code and infrastructure to detect vulnerabilities	DevOps engineer
DevOps and security: Analyze code for vulnerabilities before deployment	Security engineer
DevOps and security: Enforce security policies by preventing code with critical vulnerabilities from being merged	Security engineer
DevOps and security: Implement role-base d access control (RBAC) to restrict access to sensitive systems and data and to make sure that only authorized personnel can access critical resources	Security engineer
DevOps and security: Adjust access controls based on roles and responsibilities by adapting to changes in the team structure	DevOps engineer
DevOps and security: Test running applicati ons for security vulnerabilities in near real time by simulating attacks on the production environment	Security engineer
DevOps and security: Continuously monitor deployed applications for security vulnerabi lities	DevOps engineer
DevOps and security: Schedule regular vulnerability scans across all environments to identify and address security weaknesses	Security engineer
DevOps and security: Apply patches and updates based on vulnerability scan results to help maintain secure systems	DevOps engineer

Subcapability: Use case	Persona
Application performance monitoring: Continuously monitor application performan ce in near real time to detect and diagnose performance issues before they affect users	Site reliability engineer
Application performance monitoring: Detect performance anomalies, such as sudden spikes in response times or increased error rates, and initiate alerts	DevOps engineer
Application performance monitoring: Trace requests as they propagate through a distribut ed system to identify performance bottlenec ks and latency issues	DevOps engineer
Application performance monitoring: Use distributed tracing to pinpoint the exact service or component that is responsible for failures or performance degradation	DevOps engineer
Log aggregation and analytics: Aggregate logs from multiple sources into a centralized system for easy searching and analysis in order to identify trends and issues	Site reliability engineer
Log aggregation and analytics: Implement automated log parsing to extract relevant information and detect patterns or anomalies that might indicate issues	DevOps engineer
Log aggregation and analytics: Collect and visualize key performance metrics	Site reliability engineer
Log aggregation and analytics: Monitor metrics against predefined service-level agreements (SLAs)	Product manager

Subcapability: Use case	Persona
AI operations: Detect incidents, analyze root causes, and initiate corrective actions without human intervention	DevOps engineer
AI operations: Predict future resource demands and optimize capacity planning in order to avoid outages	Site reliability engineer
Continuous improvement: Monitor real user interactions with the application to gather insights about performance and identify areas for improvement	UX designer
Continuous improvement: Track application performance across different geographical regions to ensure consistent user experience globally	Product manager
Dashboard monitoring: Create customizable dashboards to visualize critical metrics, logs, and traces in near real time in order to provide a comprehensive view of system health	Site reliability engineer
Dashboard monitoring: Create dashboard s for different teams (such as development, operations, and product teams) to provide relevant insights based on their focus areas	DevOps engineer
Performance insights: Conduct detailed analysis of application performance to identify inefficiencies and optimize code or infrastru cture	Software developer

Subcapability: Use case	Persona
Performance insights: Use performance insights to iteratively improve application performance and optimize the user experience over time	Product manager

Generative AI use cases for operation and maintenance

After software is deployed, the focus shifts to operation and maintenance. Generative AI can enhance traditional approaches by providing more proactive and efficient system management. AI-powered operations tools continuously monitor system performance and predict potential issues before they affect users. They perform automated root cause analysis when problems occur, which significantly reduces the mean time to resolution. AI also optimizes system performance in near real time. It automatically adjusts configurations based on changing load patterns and user behaviors. For example, an operations team might use an AI assistant to generate predictive maintenance schedules, automatically identify components that are likely to fail, and suggest preemptive actions. The AI could also help with capacity planning by analyzing usage trends and predicting future resource needs with high accuracy.

The following table shows operation and maintenance use cases that you can enhance with generative AI and the persona responsible for those use cases.

Subcapability: Use case	Persona
Incident management: Manage incidents in near real time by integrating monitoring tools with chat platforms so that teams can detect, discuss, and resolve issues directly within the chat environment	Site reliability engineer
Incident management: Allow teams to initiate deployments, run scripts, and run commands directly from the chat interface, which streamlines operations	DevOps engineer

Subcapability: Use case	Persona
Code upgrades: Upgrade code dependenc ies and libraries to reduce manual effort and make sure that the codebase stays up to date with the latest versions	Software developer
Code optimization: Review code for optimizat ion opportunities	Software developer
Code optimization: Identify bottlenecks in the code and refactor or optimize the code to enhance performance	Software developer
Technical debt management: Log technical debt as part of the development process	Product manager
Technical debt management: Prioritize and address technical debt based on impact, risk, and cost, and integrate it into the regular sprint planning process	Software developer
Technical debt management: Reduce technical debt in existing application code	Software developer
Change management: Implement a change approval process that makes sure that all code changes are reviewed, tested, and approved by the necessary stakeholders before deployment	Change manager
Change management: Perform impact analysis of proposed changes	DevOps engineer
Reverse engineering: Analyze and understand the structure and behavior of legacy code	Solutions architect
Reverse engineering: Explain existing code and generate documentation	Software developer

Subcapability: Use case	Persona
Code modernization: Translate code from one programming language to another	Software developer
Code modernization: Modernize legacy code into the latest programming language	Software developer
Performance optimization: Continuously monitor and tune system performance by optimizing resource allocation, load balancing, and reconfiguring the application	Site reliability engineer
Performance optimization: Identify and refactor code that is causing performance degradation in order to improve speed and system responsiveness	Software developer

Use cases for generative AI assistants in software development

The AI assistant capability is at the heart of the generative AI-powered development experience. This intelligent, context-aware system serves as a virtual collaborator for all team members across the entire SDLC. Imagine a developer working on a complex piece of code. They can simply ask the AI assistant for help, and it can provide relevant code snippets, explain intricate algorithms, or even suggest optimizations based on the current context and best practices. The AI assistant can help an ITOps manager understand a standard operating procedure based on internal documents. By providing instant, contextual support, AI assistants significantly reduce cognitive load on team members. This helps them focus on higher-level problem-solving and creative tasks. This capability acts as a force multiplier that enhances productivity and quality across all stages of software development.

The following table shows use cases that you can enhance with AI assistants and the benefited persona.

Use case	Persona
Provide instant assistance to development team by answering questions, such as about requirements, architectures, and standard operating procedures	Software development team
Search or retrieve excerpts from extensive documentation or generate summaries by using natural language queries	Software development team
Summarize long technical documents, such as requirement documents, architecture design documentations, and internal processes	Software development team
Maintain a library of prompts that the team can use for common tasks	Software development team
Seamlessly integrate generative AI into existing tools and systems	Software development team
Automate tasks across various platforms, tools, and internal systems	Software development team
Create a centralized repository of knowledge , including best practices, project-specific information, and team knowledge, that is accessible to all team members	Software development team
Retrieve relevant knowledge from the repository based on the context of the task	Software development team
Perform automated code reviews, root cause analysis, suggest improvements, detect potential bugs, and perform troubleshooting	Software developer, DevOps engineer, and site reliability engineer

Use case	Persona
Analyze performance data to identify trends and patterns that can inform decisions about performance optimization	Site reliability engineer
Provide recommendations for improving efficiency, reducing complexity, and enhancing security	Software developer
Suggest optimizations for cloud resource usage, such as scaling recommendations or cost-saving strategies	Software developer, DevOps engineer, site reliability engineer, and solutions architect
Generate new content, such as documentation based on code, user guides, or product feature releases	Software development team

Generative AI use cases for analytics and insights

The analytics and insights capability helps convert vast amounts of data into actionable insights that drive decision making and continuous improvement. By using generative AI, this capability processes data from various sources, including code repositories, project management tools, and team collaboration platforms, to provide a holistic view of the development process and team productivity. Generative AI goes beyond traditional metrics in order to offer predictive and prescriptive analytics. It can forecast potential issues and suggest targeted improvements. For instance, it can analyze patterns in code commits, bug resolution rates, and feature delivery velocity in order to identify high-performing teams, pinpoint bottlenecks, and suggest process optimizations. Moreover, it can provide insights into team dynamics and individual performance. These insights help leaders make data-driven decisions about workload distribution, training needs, and team composition. By presenting these insights through interactive dashboards, the capability empowers stakeholders at all levels to make informed decisions, optimize processes, and continuously enhance team productivity, which leads to faster delivery of high-quality software.

The following table shows analytics use cases that you can enhance with generative AI and the persona responsible for those use cases.

Use case	Persona
Monitor individual and team productivity	Development manager
Analyze productivity trends to detect potential burnout so that you can take proactive measures to maintain team well-being and productivity	Development manager
Track how often code changes are deployed to production to gauge the speed and agility of the development process	Product manager
Analyze deployment frequency data to identify periods of low deployment activity that might indicate process inefficiencies or resource constraints	Product manager
Measure the time between code commit to deployment in order to identify opportuni ties to streamline the development and deployment processes	Development manager
Track the percentage of deployments that result in failures that require immediate remediation in order to assess the reliability of the release process	Site reliability engineer
Use change failure rate metrics to identify areas of code that frequently cause issues in order to guide targeted refactoring and testing efforts	Software developer
Monitor how long it takes to restore service after an outage or incident so that you can reduce downtime and improve the overall system resilience	Site reliability engineer

Use case	Persona
Analyze trends in restoration times to enhance incident response processes and drive faster recovery from system failures	DevOps engineer
Create a customized dashboard that aggregates key metrics, such as deployment frequency, lead time, and change failure rate, in order to provide a comprehensive view of development and operational health	Product manager
Create dashboards that are tailored to the needs of different teams in order to provide focused insights into their specific areas of responsibility, such as development, operations, or business	Product manager
Track business key performance indicators (KPIs), such as revenue impact, customer satisfaction, and market share, in order to align development efforts with broader business objectives	Product manager
Analyze the impact of new features on business KPIs to assess their success and guide future product development	Business analyst
Monitor code quality metrics, such as code complexity, test coverage, and bug density, in order to make sure that the codebase remains maintainable and secure	Software developer
Identify areas of the codebase that require refactoring in order to drive long-term sustainability and reduce technical debt	Solutions architect

Generative AI use cases for knowledge management

In any software development organization, knowledge is a critical asset. The knowledge management capability, powered by generative AI, enhances how this asset is captured, organized, and used. Traditional knowledge management systems often contain too much information, contain outdated content, or are difficult to search in order to quickly find relevant information.

Generative AI addresses these challenges head-on. It automatically generates, and updates documentation based on code changes, conversations, and project artifacts. This makes sure that knowledge bases remain current without requiring manual effort from team members. More importantly, AI makes this knowledge accessible in intuitive ways. Team members can ask questions in natural language, and the AI can provide relevant answers. The AI can draw from a variety of sources, such as official documentation, code comments, discussion threads, and even external resources. For example, a new team member trying to understand a specific component could ask the AI, "How does the authentication module work?" The AI would then provide a concise explanation and links to relevant code sections, architectural diagrams, and recent changes. It could even tailor this information based on the team member's role and level of expertise.

This capability accelerates onboarding, reduces repetitive questions, and promotes knowledge sharing across the organization. It helps preserve institutional knowledge, making it easier for teams to maintain and evolve complex systems over time.

The following table shows knowledge management use cases that you can enhance with generative AI and the persona responsible for those use cases.

Use case	Persona
Create a unified platform that makes it easy to access all project-related knowledge	Software development team
Capture knowledge from various development activities	Software development team
Provide advanced search functionality to quickly find relevant knowledge within a repository	Software development team

Use case	Persona
Personalize learning modules and pathways for the team	Software development team

Generative AI use cases for extensibility

Extensibility enables seamless integration with existing tools and workflows while allowing organizations to tailor the AI system to their specific needs. This capability provides robust APIs, SDKs, and customizable interfaces that facilitate the integration of AI functionalities into popular development and project management tools. For instance, organizations can enhance Jira with AI-powered features for automated ticket prioritization, effort estimation, and sprint planning. You can augment Jenkins pipelines with AI for intelligent build optimization and predictive test selection.

Additionally, extensibility allows for deep integration with integrated development environments (IDEs), version control systems, and code review platforms. The AI can help code, automate code reviews, and generate contextual documentation.

The capability also supports training and fine-tuning AI models on organization-specific data. This helps the AI understand company-specific coding patterns, architectural preferences, and domain knowledge. The results is more relevant and context-aware assistance across all integrated tools. By providing this level of flexibility and integration, extensibility ensures that the AI-powered development experience evolves with the organization. It can adapt to changing technologies and business needs while seamlessly enhancing existing toolchains and workflows.

The following table shows extensibility use cases that you can enhance with generative AI and the persona responsible for those use cases.

Use case	Persona
Integrate third-party tools into the developme nt environment	DevOps engineer
Create custom automation workflows that are tailored to team's unique development process	DevOps engineer

Use case	Persona
Connect to various APIs and services	DevOps engineer
Create connectors for cross-platform tools	DevOps engineer

Best practices for using generative AI in software development

This section describes best practices for integrating generative AI into the software development lifecycle (SDLC). From implementing seamless toolchains and DevSecOps pipelines to fostering collaboration and automating repetitive tasks, these guidelines help you harness the power of AI to enhance your development processes and experiences. By following these best practices, software development teams can unlock new levels of efficiency, innovation, and quality in their work.

This section discusses the following best practices:

- Implementing a seamless, end-to-end integrated toolchain
- Implementing an end-to-end CI/CD pipeline for DevSecOps
- Adopting collaborative tools and practices
- Automating repetitive tasks
- Regularly reviewing and iterating on the development experience
- Adopting effective project management practices
- Implementing knowledge management
- Providing extensibility and customization
- Optimizing for operations
- Using data-driven insights
- Adopting a platform-based approach

Implementing a seamless, end-to-end integrated toolchain

Implementing a seamless, end-to-end integrated toolchain is a foundational best practice for creating a generative AI-powered development experience. The core idea is to establish a cohesive ecosystem of tools and platforms that your software teams can use across the entire SDLC. The team can use the toolchain to plan, ideate, code, build, test, deploy, and manage ongoing operations. By integrating generative AI capabilities into this toolchain, you make sure that AI assistance is available at every stage. This integration reduces or eliminates manual handoffs, reduces context switching, and helps data and artifacts flow smoothly between different development phases. For example, AI-generated code snippets from your integrated development

environment (IDE) can seamlessly flow into your version control system, and AI-powered analytics from your deployment platform can inform your project management tools. This creates a continuous feedback loop that improves your development process.

Implementing an end-to-end CI/CD pipeline for DevSecOps

To build upon this integrated toolchain, implement an end-to-end continuous integration and continuous deployment (CI/CD) pipeline for DevSecOps. This AI-powered pipeline is a critical component that streamlines your software delivery processes. It helps you release new applications and updates more quickly and reliably. By embedding security practices throughout the entire SDLC, you can identify and address vulnerabilities much earlier, which reduces the overall cost and risk. The pipeline should incorporate AI at every stage, from continuous integration and testing to security checks and deployment. For instance, you can use AI to analyze code commits in near real time so that you can predict potential integration issues before they occur. In the CI/CD pipeline, you can also use generative AI to automatically update security policies based on the latest threat intelligence.

Adopting collaborative tools and practices

As you enhance your development infrastructure, don't forget the human element. Software development is inherently a collaborative endeavor. It involves cross-functional teams composed of developers, designers, product managers, Scrum Masters, business analysts, and other stakeholders. These individuals work collectively to bring ideas to fruition. By using modern collaborative tools and fostering a culture of open communication and knowledge sharing, you can significantly enhance the productivity and effectiveness of your software development teams. In your AI-powered software development experience, these tools take on new dimensions. You can integrate AI into collaboration platforms to facilitate more effective communication and knowledge sharing among team members. AI assistants can answer common questions, summarize discussions, or even mediate conflicts. Generative AI can enhance code review processes by automatically suggesting improvements or identifying potential issues. Furthermore, you can use AI to create dynamic, context-aware documentation that updates in near real time as the project evolves so that all team members have access to the most current and relevant information.

Automating repetitive tasks

By using generative AI to handle routine, time-consuming activities, you free your software teams to focus on high-value, creative work that drives innovation and delivers business impact. Examples

of repetitive tasks include generating boilerplate code, creating test data, writing documentation, or even drafting initial project plans. By offloading these tasks to AI, team members can focus on more creative and strategic work. For instance, AI-powered code-completion tools can significantly speed up the coding process by suggesting relevant code snippets based on context and coding patterns. Similarly, generative AI can automatically create and update technical documentation as code changes. This keeps the documentation current and reduces the manual effort typically required for this task. In testing, AI can generate comprehensive test cases based on requirements and code analysis, which improves test coverage and reduces the likelihood of overlooked edge cases. By intelligently automating these repetitive tasks, generative AI accelerates development timelines, improves consistency, and reduces human error. The result is higher quality software outputs.

Regularly reviewing and iterating on the development experience

Your software development experience itself should be treated as a product that requires ongoing refinement. This involves establishing a systematic process for regularly reviewing and iterating on all aspects of the development lifecycle, tools, and practices. Perform periodic assessments of the entire toolchain, workflows, and processes. Gather feedback from all team members across various roles, including product managers, designers, architects, developers, testers, and operations personnel. Ask them to identify pain points, bottlenecks, and opportunities for enhancement. For example, teams might conduct quarterly reviews of their CI/CD pipeline performance and analyze metrics such as build times, deployment frequency, and error rates in order to identify areas for optimization. Because generative AI capabilities continue to evolve rapidly, it's crucial to consistently evaluate new AI-powered tools and features that might further streamline workflows or augment capabilities across all roles in the SDLC.

Adopting effective project management practices

To orchestrate your complex software development efforts effectively, adopt AI-augmented project management practices. In this context, effective project management goes beyond traditional methodologies. It embraces AI-augmented approaches that enhance planning, execution, and monitoring across the entire SDLC. Agile frameworks promote flexibility, collaboration, and rapid iteration, and you can use generative AI to optimize these processes. For instance, generative AI can analyze historical project data for more accurate estimates, automatically generate and prioritize user stories based on business objectives and customer feedback, and provide intelligent

insights into team performance. AI-powered project management tools can predict potential roadblocks and suggest optimal task assignments based on team members' skills and workloads. By integrating AI-powered capabilities into project management practices, you can achieve greater visibility, make data-driven decisions faster, and make sure that team members are aligned and working efficiently toward common goals.

Implementing knowledge management

As your AI-powered software development experience matures, implement a robust knowledge management system. A robust knowledge management system helps you capture, organize, and grant access to valuable insights, best practices, and solutions. All team members across the SDLC should have easy access to the system. Use generative AI to create dynamic, intelligent knowledge bases that evolve with your organization. For instance, AI can automatically generate and update documentation based on code changes, conversations, and project artifacts so that information remains current without manual intervention. Generative AI can also power intelligent search capabilities and help team members quickly find relevant information by using natural language queries, even if they don't know the exact terminology. Furthermore, generative AI can proactively surface relevant information to team members based on their current tasks or challenges. It acts as a virtual mentor that enhances decision making and problem solving across all roles. By implementing an AI-powered knowledge management system, you can break down silos, accelerate onboarding, reduce redundant work, and foster a culture of continuous learning and innovation throughout your entire software development team.

Providing extensibility and customization

To maximize the benefits of generative AI in software development, make sure that your AI-powered tools and platforms are extensible and customizable. This helps you to tailor AI capabilities to your specific needs, workflows, and technology stacks. For example, you can fine tune AI models on your own codebases and documentation, create custom AI-powered tools for specific tasks, or integrate AI capabilities into existing tools and processes. This extensibility helps you evolve the AI-powered development experience to meet the organization's changing needs. It also helps you optimize the experience for specific domains or project types.

Optimizing for operations

Generative AI plays a crucial role in optimizing software operations and maintenance. Optimize for operations by integrating AI capabilities into your operational tools and processes. For instance, use

generative AI to analyze log data in near real time, predict potential system failures, and automate routine maintenance tasks. Generative AI can also help with root cause analysis by correlating events across complex distributed systems. This improves system reliability, reduces downtime, and frees your operations teams to focus on more strategic initiatives.

Using data-driven insights

Use data-driven insights throughout your AI-powered development journey. Implement systems to collect, analyze, and act upon data from all stages of the SDLC. This includes code metrics, test results, deployment data, user feedback, and operational performance. Use generative AI to uncover patterns and insights that might not be apparent to human observers. Then, feed these insights back into your development process to inform everything from architectural decisions to feature prioritization.

Adopting a platform-based approach

To fully realize the benefits of generative AI in software development, adopt a platform-based approach. Create a comprehensive, integrated platform that incorporates AI capabilities across all aspects of the SDLC. The platform should provide a consistent user experience, centralized management and data, and seamless integration between different tools and processes. This makes AI benefits uniformly available across your organization, reduces the overhead of managing multiple and disparate AI tools, and provides a foundation for continuous improvement and expansion of AI capabilities.

Measuring the success of generative AI in software development

To effectively measure the effect of implementing a generative AI-powered software development experience, you need to establish a comprehensive set of metrics that span across various dimensions of your software development lifecycle (SDLC). These metrics should capture immediate improvements in efficiency and productivity and also reflect long-term gains in software quality, team satisfaction, and business value.

Do the following to effectively use the recommended metrics in this section:

1. **Establish baselines** – Before you dive into implementing your AI-powered development experience, take time to gather comprehensive data about your current performance across these metrics. This provides a clear starting point and helps you make meaningful comparisons later.
2. **Set realistic targets** – With your baselines in hand, set achievable improvement targets for each metric. Be ambitious but realistic. Remember that sustainable progress is often incremental.
3. **Implement continuous monitoring** – Use automated tools to constantly collect and analyze data for these metrics in your environment. Near real-time monitoring helps you monitor progress and quickly identify any issues or opportunities.
4. **Conduct regular reviews** – Schedule quarterly or biannual review sessions where you and your team thoroughly assess your progress against the targets. Use these sessions to identify areas for further improvement and celebrate your successes.
5. **Iterate and adjust** – Based on the insights you've gained, continuously refine your generative AI implementation and adjust targets as necessary.

This section describes the following categories of metrics:

Deployment velocity

Consider measuring the following deployment velocity metrics.

Metric	Description
Time to market	Measure the reduction in time from idea conception to production deployment
Sprint velocity	Track the increase in story points completed per sprint by your teams
Code commit frequency	Monitor the increase in code commits, which indicates accelerating development cycles
Pull request resolution time	Assess the decrease in time taken to review and merge code changes in your repositories
Release velocity	Measure the increase in the number of releases per quarter or year

Code quality

Consider measuring the following code quality metrics.

Metric	Description
Defect density	Measure the reduction in software bugs
Code coverage	Track the increase in test coverage percentage across your codebase
Technical debt	Monitor the decrease in identified technical debt over time
Static code analysis scores	Assess improvements in code quality based on your automated analysis tools

Operational efficiency

Consider measuring the following operational efficiency metrics.

Metric	Description
Deployment frequency	Measure the increase in the number of successful deployments
Mean time to recovery (MTTR)	Track the reduction in the amount of time it takes to recover from system failures
Change failure rate	Monitor the decrease in the percentage of changes that result in failures in your deployments

Team productivity and satisfaction

Consider measuring the following team productivity and satisfaction metrics.

Metric	Description
Productivity improvement	Monitor the increase in productivity percentag e for each task
Satisfaction score	Conduct regular surveys to gauge improveme nt in your team's morale and job satisfaction
Knowledge sharing efficiency	Measure the reduction in time your team spends searching for information or asking repetitive questions
On-boarding time	Track the decrease in time required for new team members to become productive

Business impact

Consider measuring the following business impact metrics.

Metric	Description
Feature adoption rate	Measure the increase in user engagement with new features you've released
Customer satisfaction score	Track improvements in your user feedback and ratings
Revenue impact (direct and indirect)	Assess the increase in revenue attributed to increased release velocity or increased productivity

Conclusion

This strategy document provides an overview of a generative AI-powered software development experience. It explores the five dimensions in the 5-I framework—Investigate, Integrate, Interact, Iterate, and Impact. These dimensions provide a strategic roadmap for integrating generative AI across the entire software development lifecycle (SDLC). It also describes the foundational capabilities that are required to successfully implement this framework. The capabilities span areas like project management, DevSecOps, AI assistants, knowledge management, and more. It provides best practices to consider when integrating generative AI, and it helps you use metrics to measure the impact that generative AI has on your software development experience.

The integration of generative AI into software development processes represents a paradigm shift that has the potential to accelerate innovation, improve quality, and enhance productivity. However, it's important to recognize that this is not a one-time implementation. It is an ongoing evolution that requires sustained effort and continuous refinement.

As you embark on this journey, we recommend that you start with a thorough assessment of your organization's current capabilities and readiness. The AWS Assessment Tool is an AI-powered software development assessment tool that can help you identify priority areas and create a tailored implementation roadmap.

Resources

Once you've identified key priority areas, following resources can help you implement your roadmap:

AWS documentation

- Automate AWS infrastructure operations by using Amazon Bedrock (AWS Prescriptive Guidance)
- Best practices with Amazon Q Developer for in-line and assistant code generation (AWS Prescriptive Guidance)
- Develop a fully automated chat-based assistant by using Amazon Bedrock agents and knowledge bases (AWS Prescriptive Guidance)
- Transforming application development and maintenance operating models on AWS with generative AI (AWS Prescriptive Guidance)

- Use Amazon Q Developer as a coding assistant to increase your productivity (AWS Prescriptive Guidance)

AWS blog posts and tutorials

- Amazon Q blog posts
- Accelerate your Software Development Lifecycle with Amazon Q (AWS blog post)
- Building an AWS Solutions Architect AI Agent: Leveraging Amazon Bedrock for Automated Architecture and Deployment (AWS video)
- Generative AI-powered technology operations (AWS blog post)
- Modernize your Java application with Amazon Q Developer (AWS blog post)
- Use Amazon Bedrock to generate, evaluate, and understand code in your software development pipeline (AWS blog post)

Document history

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Initial publication	Not applicable	April 18, 2025

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Numbers

7 Rs

Seven common migration strategies for moving applications to the cloud. These strategies build upon the 5 Rs that Gartner identified in 2011 and consist of the following:

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.
- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.
- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.
- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.
- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. You migrate servers from an on-premises platform to a cloud service for the same platform. Example: Migrate a Microsoft Hyper-V application to AWS.
- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there's no business justification for migrating them.

- Retire – Decommission or remove applications that are no longer needed in your source environment.

A

ABAC

See attribute-based access control.

abstracted services

See managed services.

ACID

See atomicity, consistency, isolation, durability.

active-active migration

A database migration method in which the source and target databases are kept in sync (by using a bidirectional replication tool or dual write operations), and both databases handle transactions from connecting applications during migration. This method supports migration in small, controlled batches instead of requiring a one-time cutover. It's more flexible but requires more work than active-passive migration.

active-passive migration

A database migration method in which the source and target databases are kept in sync, but only the source database handles transactions from connecting applications while data is replicated to the target database. The target database doesn't accept any transactions during migration.

aggregate function

A SQL function that operates on a group of rows and calculates a single return value for the group. Examples of aggregate functions include `SUM` and `MAX`.

AI

See artificial intelligence.

AIOps

See artificial intelligence operations.

anonymization

The process of permanently deleting personal information in a dataset. Anonymization can help protect personal privacy. Anonymized data is no longer considered to be personal data.

anti-pattern

A frequently used solution for a recurring issue where the solution is counter-productive, ineffective, or less effective than an alternative.

application control

A security approach that allows the use of only approved applications in order to help protect a system from malware.

application portfolio

A collection of detailed information about each application used by an organization, including the cost to build and maintain the application, and its business value. This information is key to the portfolio discovery and analysis process and helps identify and prioritize the applications to be migrated, modernized, and optimized.

artificial intelligence (AI)

The field of computer science that is dedicated to using computing technologies to perform cognitive functions that are typically associated with humans, such as learning, solving problems, and recognizing patterns. For more information, see What is Artificial Intelligence?

artificial intelligence operations (AIOps)

The process of using machine learning techniques to solve operational problems, reduce operational incidents and human intervention, and increase service quality. For more information about how AIOps is used in the AWS migration strategy, see the operations integration guide.

asymmetric encryption

An encryption algorithm that uses a pair of keys, a public key for encryption and a private key for decryption. You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted.

atomicity, consistency, isolation, durability (ACID)

A set of software properties that guarantee the data validity and operational reliability of a database, even in the case of errors, power failures, or other problems.

attribute-based access control (ABAC)

The practice of creating fine-grained permissions based on user attributes, such as department, job role, and team name. For more information, see ABAC for AWS in the AWS Identity and Access Management (IAM) documentation.

authoritative data source

A location where you store the primary version of data, which is considered to be the most reliable source of information. You can copy data from the authoritative data source to other locations for the purposes of processing or modifying the data, such as anonymizing, redacting, or pseudonymizing it.

Availability Zone

A distinct location within an AWS Region that is insulated from failures in other Availability Zones and provides inexpensive, low-latency network connectivity to other Availability Zones in the same Region.

AWS Cloud Adoption Framework (AWS CAF)

A framework of guidelines and best practices from AWS to help organizations develop an efficient and effective plan to move successfully to the cloud. AWS CAF organizes guidance into six focus areas called perspectives: business, people, governance, platform, security, and operations. The business, people, and governance perspectives focus on business skills and processes; the platform, security, and operations perspectives focus on technical skills and processes. For example, the people perspective targets stakeholders who handle human resources (HR), staffing functions, and people management. For this perspective, AWS CAF provides guidance for people development, training, and communications to help ready the organization for successful cloud adoption. For more information, see the AWS CAF website and the AWS CAF whitepaper.

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

B

bad bot

A bot that is intended to disrupt or cause harm to individuals or organizations.

BCP

See business continuity planning.

behavior graph

A unified, interactive view of resource behavior and interactions over time. You can use a behavior graph with Amazon Detective to examine failed logon attempts, suspicious API calls, and similar actions. For more information, see Data in a behavior graph in the Detective documentation.

big-endian system

A system that stores the most significant byte first. See also endianness.

binary classification

A process that predicts a binary outcome (one of two possible classes). For example, your ML model might need to predict problems such as “Is this email spam or not spam?" or "Is this product a book or a car?"

bloom filter

A probabilistic, memory-efficient data structure that is used to test whether an element is a member of a set.

blue/green deployment

A deployment strategy where you create two separate but identical environments. You run the current application version in one environment (blue) and the new application version in the other environment (green). This strategy helps you quickly roll back with minimal impact.

bot

A software application that runs automated tasks over the internet and simulates human activity or interaction. Some bots are useful or beneficial, such as web crawlers that index information on the internet. Some other bots, known as *bad bots*, are intended to disrupt or cause harm to individuals or organizations.

botnet

Networks of bots that are infected by malware and are under the control of a single party, known as a *bot herder* or *bot operator*. Botnets are the best-known mechanism to scale bots and their impact.

branch

A contained area of a code repository. The first branch created in a repository is the *main branch*. You can create a new branch from an existing branch, and you can then develop features or fix bugs in the new branch. A branch you create to build a feature is commonly referred to as a *feature branch*. When the feature is ready for release, you merge the feature branch back into the main branch. For more information, see About branches (GitHub documentation).

break-glass access

In exceptional circumstances and through an approved process, a quick means for a user to gain access to an AWS account that they don't typically have permissions to access. For more information, see the Implement break-glass procedures indicator in the AWS Well-Architected guidance.

brownfield strategy

The existing infrastructure in your environment. When adopting a brownfield strategy for a system architecture, you design the architecture around the constraints of the current systems and infrastructure. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

buffer cache

The memory area where the most frequently accessed data is stored.

business capability

What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

business continuity planning (BCP)

A plan that addresses the potential impact of a disruptive event, such as a large-scale migration, on operations and enables a business to resume operations quickly.

C

CAF

See AWS Cloud Adoption Framework.

canary deployment

The slow and incremental release of a version to end users. When you are confident, you deploy the new version and replace the current version in its entirety.

CCoE

See Cloud Center of Excellence.

CDC

See change data capture.

change data capture (CDC)

The process of tracking changes to a data source, such as a database table, and recording metadata about the change. You can use CDC for various purposes, such as auditing or replicating changes in a target system to maintain synchronization.

chaos engineering

Intentionally introducing failures or disruptive events to test a system's resilience. You can use AWS Fault Injection Service (AWS FIS) to perform experiments that stress your AWS workloads and evaluate their response.

CI/CD

See continuous integration and continuous delivery.

classification

A categorization process that helps generate predictions. ML models for classification problems predict a discrete value. Discrete values are always distinct from one another. For example, a model might need to evaluate whether or not there is a car in an image.

client-side encryption

Encryption of data locally, before the target AWS service receives it.

Cloud Center of Excellence (CCoE)

A multi-disciplinary team that drives cloud adoption efforts across an organization, including developing cloud best practices, mobilizing resources, establishing migration timelines, and leading the organization through large-scale transformations. For more information, see the CCoE posts on the AWS Cloud Enterprise Strategy Blog.

cloud computing

The cloud technology that is typically used for remote data storage and IoT device management. Cloud computing is commonly connected to edge computing technology.

cloud operating model

In an IT organization, the operating model that is used to build, mature, and optimize one or more cloud environments. For more information, see Building your Cloud Operating Model.

cloud stages of adoption

The four phases that organizations typically go through when they migrate to the AWS Cloud:

- Project – Running a few cloud-related projects for proof of concept and learning purposes
- Foundation – Making foundational investments to scale your cloud adoption (e.g., creating a landing zone, defining a CCoE, establishing an operations model)
- Migration – Migrating individual applications
- Re-invention – Optimizing products and services, and innovating in the cloud

These stages were defined by Stephen Orban in the blog post The Journey Toward Cloud-First & the Stages of Adoption on the AWS Cloud Enterprise Strategy blog. For information about how they relate to the AWS migration strategy, see the migration readiness guide.

CMDB

See configuration management database.

code repository

A location where source code and other assets, such as documentation, samples, and scripts, are stored and updated through version control processes. Common cloud repositories include GitHub or Bitbucket Cloud. Each version of the code is called a *branch*. In a microservice structure, each repository is devoted to a single piece of functionality. A single CI/CD pipeline can use multiple repositories.

cold cache

A buffer cache that is empty, not well populated, or contains stale or irrelevant data. This affects performance because the database instance must read from the main memory or disk, which is slower than reading from the buffer cache.

cold data

Data that is rarely accessed and is typically historical. When querying this kind of data, slow queries are typically acceptable. Moving this data to lower-performing and less expensive storage tiers or classes can reduce costs.

computer vision (CV)

A field of AI that uses machine learning to analyze and extract information from visual formats such as digital images and videos. For example, Amazon SageMaker AI provides image processing algorithms for CV.

configuration drift

For a workload, a configuration change from the expected state. It might cause the workload to become noncompliant, and it's typically gradual and unintentional.

configuration management database (CMDB)

A repository that stores and manages information about a database and its IT environment, including both hardware and software components and their configurations. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

conformance pack

A collection of AWS Config rules and remediation actions that you can assemble to customize your compliance and security checks. You can deploy a conformance pack as a single entity in an AWS account and Region, or across an organization, by using a YAML template. For more information, see Conformance packs in the AWS Config documentation.

continuous integration and continuous delivery (CI/CD)

The process of automating the source, build, test, staging, and production stages of the software release process. CI/CD is commonly described as a pipeline. CI/CD can help you automate processes, improve productivity, improve code quality, and deliver faster. For more information, see Benefits of continuous delivery. CD can also stand for *continuous deployment*. For more information, see Continuous Delivery vs. Continuous Deployment.

CV

See computer vision.

D

data at rest

Data that is stationary in your network, such as data that is in storage.

data classification

A process for identifying and categorizing the data in your network based on its criticality and sensitivity. It is a critical component of any cybersecurity risk management strategy because it helps you determine the appropriate protection and retention controls for the data. Data classification is a component of the security pillar in the AWS Well-Architected Framework. For more information, see Data classification.

data drift

A meaningful variation between the production data and the data that was used to train an ML model, or a meaningful change in the input data over time. Data drift can reduce the overall quality, accuracy, and fairness in ML model predictions.

data in transit

Data that is actively moving through your network, such as between network resources.

data mesh

An architectural framework that provides distributed, decentralized data ownership with centralized management and governance.

data minimization

The principle of collecting and processing only the data that is strictly necessary. Practicing data minimization in the AWS Cloud can reduce privacy risks, costs, and your analytics carbon footprint.

data perimeter

A set of preventive guardrails in your AWS environment that help make sure that only trusted identities are accessing trusted resources from expected networks. For more information, see Building a data perimeter on AWS.

data preprocessing

To transform raw data into a format that is easily parsed by your ML model. Preprocessing data can mean removing certain columns or rows and addressing missing, inconsistent, or duplicate values.

data provenance

The process of tracking the origin and history of data throughout its lifecycle, such as how the data was generated, transmitted, and stored.

data subject

An individual whose data is being collected and processed.

data warehouse

A data management system that supports business intelligence, such as analytics. Data warehouses commonly contain large amounts of historical data, and they are typically used for queries and analysis.

database definition language (DDL)

Statements or commands for creating or modifying the structure of tables and objects in a database.

database manipulation language (DML)

Statements or commands for modifying (inserting, updating, and deleting) information in a database.

DDL

See database definition language.

deep ensemble

To combine multiple deep learning models for prediction. You can use deep ensembles to obtain a more accurate prediction or for estimating uncertainty in predictions.

deep learning

An ML subfield that uses multiple layers of artificial neural networks to identify mapping between input data and target variables of interest.

defense-in-depth

An information security approach in which a series of security mechanisms and controls are thoughtfully layered throughout a computer network to protect the confidentiality, integrity, and availability of the network and the data within. When you adopt this strategy on AWS, you add multiple controls at different layers of the AWS Organizations structure to help secure resources. For example, a defense-in-depth approach might combine multi-factor authentication, network segmentation, and encryption.

delegated administrator

In AWS Organizations, a compatible service can register an AWS member account to administer the organization's accounts and manage permissions for that service. This account is called the *delegated administrator* for that service. For more information and a list of compatible services, see Services that work with AWS Organizations in the AWS Organizations documentation.

deployment

The process of making an application, new features, or code fixes available in the target environment. Deployment involves implementing changes in a code base and then building and running that code base in the application's environments.

development environment

See environment.

detective control

A security control that is designed to detect, log, and alert after an event has occurred. These controls are a second line of defense, alerting you to security events that bypassed the preventative controls in place. For more information, see Detective controls in *Implementing security controls on AWS*.

development value stream mapping (DVSM)

A process used to identify and prioritize constraints that adversely affect speed and quality in a software development lifecycle. DVSM extends the value stream mapping process originally designed for lean manufacturing practices. It focuses on the steps and teams required to create and move value through the software development process.

digital twin

A virtual representation of a real-world system, such as a building, factory, industrial equipment, or production line. Digital twins support predictive maintenance, remote monitoring, and production optimization.

dimension table

In a star schema, a smaller table that contains data attributes about quantitative data in a fact table. Dimension table attributes are typically text fields or discrete numbers that behave like text. These attributes are commonly used for query constraining, filtering, and result set labeling.

disaster

An event that prevents a workload or system from fulfilling its business objectives in its primary deployed location. These events can be natural disasters, technical failures, or the result of human actions, such as unintentional misconfiguration or a malware attack.

disaster recovery (DR)

The strategy and process you use to minimize downtime and data loss caused by a disaster. For more information, see Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

See database manipulation language.

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

See disaster recovery.

drift detection

Tracking deviations from a baselined configuration. For example, you can use AWS CloudFormation to detect drift in system resources, or you can use AWS Control Tower to detect changes in your landing zone that might affect compliance with governance requirements.

DVSM

See development value stream mapping.

E

EDA

See exploratory data analysis.

EDI

See electronic data interchange.

edge computing

The technology that increases the computing power for smart devices at the edges of an IoT network. When compared with cloud computing, edge computing can reduce communication latency and improve response time.

electronic data interchange (EDI)

The automated exchange of business documents between organizations. For more information, see What is Electronic Data Interchange.

encryption

A computing process that transforms plaintext data, which is human-readable, into ciphertext.

encryption key

A cryptographic string of randomized bits that is generated by an encryption algorithm. Keys can vary in length, and each key is designed to be unpredictable and unique.

endianness

The order in which bytes are stored in computer memory. Big-endian systems store the most significant byte first. Little-endian systems store the least significant byte first.

endpoint

See service endpoint.

endpoint service

A service that you can host in a virtual private cloud (VPC) to share with other users. You can create an endpoint service with AWS PrivateLink and grant permissions to other AWS accounts or to AWS Identity and Access Management (IAM) principals. These accounts or principals can connect to your endpoint service privately by creating interface VPC endpoints. For more information, see Create an endpoint service in the Amazon Virtual Private Cloud (Amazon VPC) documentation.

enterprise resource planning (ERP)

A system that automates and manages key business processes (such as accounting, MES, and project management) for an enterprise.

envelope encryption

The process of encrypting an encryption key with another encryption key. For more information, see Envelope encryption in the AWS Key Management Service (AWS KMS) documentation.

environment

An instance of a running application. The following are common types of environments in cloud computing:

- development environment – An instance of a running application that is available only to the core team responsible for maintaining the application. Development environments are used to test changes before promoting them to upper environments. This type of environment is sometimes referred to as a *test environment*.
- lower environments – All development environments for an application, such as those used for initial builds and tests.
- production environment – An instance of a running application that end users can access. In a CI/CD pipeline, the production environment is the last deployment environment.
- upper environments – All environments that can be accessed by users other than the core development team. This can include a production environment, preproduction environments, and environments for user acceptance testing.

epic

In agile methodologies, functional categories that help organize and prioritize your work. Epics provide a high-level description of requirements and implementation tasks. For example, AWS CAF security epics include identity and access management, detective controls, infrastructure security, data protection, and incident response. For more information about epics in the AWS migration strategy, see the program implementation guide.

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

The process of analyzing a dataset to understand its main characteristics. You collect or aggregate data and then perform initial investigations to find patterns, detect anomalies, and check assumptions. EDA is performed by calculating summary statistics and creating data visualizations.

F

fact table

The central table in a star schema. It stores quantitative data about business operations. Typically, a fact table contains two types of columns: those that contain measures and those that contain a foreign key to a dimension table.

fail fast

A philosophy that uses frequent and incremental testing to reduce the development lifecycle. It is a critical part of an agile approach.

fault isolation boundary

In the AWS Cloud, a boundary such as an Availability Zone, AWS Region, control plane, or data plane that limits the effect of a failure and helps improve the resilience of workloads. For more information, see AWS Fault Isolation Boundaries.

feature branch

See branch.

features

The input data that you use to make a prediction. For example, in a manufacturing context, features could be images that are periodically captured from the manufacturing line.

feature importance

How significant a feature is for a model's predictions. This is usually expressed as a numerical score that can be calculated through various techniques, such as Shapley Additive Explanations (SHAP) and integrated gradients. For more information, see Machine learning model interpretability with AWS.

feature transformation

To optimize data for the ML process, including enriching data with additional sources, scaling values, or extracting multiple sets of information from a single data field. This enables the ML model to benefit from the data. For example, if you break down the "2021-05-27 00:15:37" date into "2021", "May", "Thu", and "15", you can help the learning algorithm learn nuanced patterns associated with different data components.

few-shot prompting

Providing an LLM with a small number of examples that demonstrate the task and desired output before asking it to perform a similar task. This technique is an application of in-context learning, where models learn from examples (*shots*) that are embedded in prompts. Few-shot prompting can be effective for tasks that require specific formatting, reasoning, or domain knowledge. See also zero-shot prompting.

FGAC

See fine-grained access control.

fine-grained access control (FGAC)

The use of multiple conditions to allow or deny an access request.

flash-cut migration

A database migration method that uses continuous data replication through change data capture to migrate data in the shortest time possible, instead of using a phased approach. The objective is to keep downtime to a minimum.

FM

See foundation model.

foundation model (FM)

A large deep-learning neural network that has been training on massive datasets of generalized and unlabeled data. FMs are capable of performing a wide variety of general tasks, such as understanding language, generating text and images, and conversing in natural language. For more information, see What are Foundation Models.

G

generative AI

A subset of AI models that have been trained on large amounts of data and that can use a simple text prompt to create new content and artifacts, such as images, videos, text, and audio. For more information, see What is Generative AI.

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

In Amazon CloudFront, an option to prevent users in specific countries from accessing content distributions. You can use an allow list or block list to specify approved and banned countries. For more information, see Restricting the geographic distribution of your content in the CloudFront documentation.

Gitflow workflow

An approach in which lower and upper environments use different branches in a source code repository. The Gitflow workflow is considered legacy, and the trunk-based workflow is the modern, preferred approach.

golden image

A snapshot of a system or software that is used as a template to deploy new instances of that system or software. For example, in manufacturing, a golden image can be used to provision software on multiple devices and helps improve speed, scalability, and productivity in device manufacturing operations.

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction

of compatibility with existing infrastructure, also known as brownfield. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries. *Detective guardrails* detect policy violations and compliance issues, and generate alerts for remediation. They are implemented by using AWS Config, AWS Security Hub CSPM, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See high availability.

heterogeneous database migration

Migrating your source database to a target database that uses a different database engine (for example, Oracle to Amazon Aurora). Heterogeneous migration is typically part of a re-architecting effort, and converting the schema can be a complex task. AWS provides AWS SCT that helps with schema conversions.

high availability (HA)

The ability of a workload to operate continuously, without intervention, in the event of challenges or disasters. HA systems are designed to automatically fail over, consistently deliver high-quality performance, and handle different loads and failures with minimal performance impact.

historian modernization

An approach used to modernize and upgrade operational technology (OT) systems to better serve the needs of the manufacturing industry. A *historian* is a type of database that is used to collect and store data from various sources in a factory.

holdout data

A portion of historical, labeled data that is withheld from a dataset that is used to train a machine learning model. You can use holdout data to evaluate the model performance by comparing the model predictions against the holdout data.

homogeneous database migration

Migrating your source database to a target database that shares the same database engine (for example, Microsoft SQL Server to Amazon RDS for SQL Server). Homogeneous migration is typically part of a rehosting or replatforming effort. You can use native database utilities to migrate the schema.

hot data

Data that is frequently accessed, such as real-time data or recent translational data. This data typically requires a high-performance storage tier or class to provide fast query responses.

hotfix

An urgent fix for a critical issue in a production environment. Due to its urgency, a hotfix is usually made outside of the typical DevOps release workflow.

hypercare period

Immediately following cutover, the period of time when a migration team manages and monitors the migrated applications in the cloud in order to address any issues. Typically, this period is 1–4 days in length. At the end of the hypercare period, the migration team typically transfers responsibility for the applications to the cloud operations team.

I

IaC

See infrastructure as code.

identity-based policy

A policy attached to one or more IAM principals that defines their permissions within the AWS Cloud environment.

idle application

An application that has an average CPU and memory usage between 5 and 20 percent over a period of 90 days. In a migration project, it is common to retire these applications or retain them on premises.

IIoT

See industrial Internet of Things.

immutable infrastructure

A model that deploys new infrastructure for production workloads instead of updating, patching, or modifying the existing infrastructure. Immutable infrastructures are inherently more consistent, reliable, and predictable than mutable infrastructure. For more information, see the Deploy using immutable infrastructure best practice in the AWS Well-Architected Framework.

inbound (ingress) VPC

In an AWS multi-account architecture, a VPC that accepts, inspects, and routes network connections from outside an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

incremental migration

A cutover strategy in which you migrate your application in small parts instead of performing a single, full cutover. For example, you might move only a few microservices or users to the new system initially. After you verify that everything is working properly, you can incrementally move additional microservices or users until you can decommission your legacy system. This strategy reduces the risks associated with large migrations.

Industry 4.0

A term that was introduced by Klaus Schwab in 2016 to refer to the modernization of manufacturing processes through advances in connectivity, real-time data, automation, analytics, and AI/ML.

infrastructure

All of the resources and assets contained within an application's environment.

infrastructure as code (IaC)

The process of provisioning and managing an application's infrastructure through a set of configuration files. IaC is designed to help you centralize infrastructure management, standardize resources, and scale quickly so that new environments are repeatable, reliable, and consistent.

industrial Internet of Things (IIoT)

The use of internet-connected sensors and devices in the industrial sectors, such as manufacturing, energy, automotive, healthcare, life sciences, and agriculture. For more information, see Building an industrial Internet of Things (IIoT) digital transformation strategy.

inspection VPC

In an AWS multi-account architecture, a centralized VPC that manages inspections of network traffic between VPCs (in the same or different AWS Regions), the internet, and on-premises networks. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

Internet of Things (IoT)

The network of connected physical objects with embedded sensors or processors that communicate with other devices and systems through the internet or over a local communication network. For more information, see What is IoT?

interpretability

A characteristic of a machine learning model that describes the degree to which a human can understand how the model's predictions depend on its inputs. For more information, see Machine learning model interpretability with AWS.

IoT

See Internet of Things.

IT information library (ITIL)

A set of best practices for delivering IT services and aligning these services with business requirements. ITIL provides the foundation for ITSM.

IT service management (ITSM)

Activities associated with designing, implementing, managing, and supporting IT services for an organization. For information about integrating cloud operations with ITSM tools, see the operations integration guide.

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

An implementation of mandatory access control (MAC) where the users and the data itself are each explicitly assigned a security label value. The intersection between the user security label and data security label determines which rows and columns can be seen by the user.

landing zone

A landing zone is a well-architected, multi-account AWS environment that is scalable and secure. This is a starting point from which your organizations can quickly launch and deploy workloads and applications with confidence in their security and infrastructure environment. For more information about landing zones, see Setting up a secure and scalable multi-account AWS environment.

large language model (LLM)

A deep learning AI model that is pretrained on a vast amount of data. An LLM can perform multiple tasks, such as answering questions, summarizing documents, translating text into other languages, and completing sentences. For more information, see What are LLMs.

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

The security best practice of granting the minimum permissions required to perform a task. For more information, see Apply least-privilege permissions in the IAM documentation.

lift and shift

See 7 Rs.

little-endian system

A system that stores the least significant byte first. See also endianness.

LLM

See large language model.

lower environments

See environment.

M

machine learning (ML)

A type of artificial intelligence that uses algorithms and techniques for pattern recognition and learning. ML analyzes and learns from recorded data, such as Internet of Things (IoT) data, to generate a statistical model based on patterns. For more information, see Machine Learning.

main branch

See branch.

malware

Software that is designed to compromise computer security or privacy. Malware might disrupt computer systems, leak sensitive information, or gain unauthorized access. Examples of malware include viruses, worms, ransomware, Trojan horses, spyware, and keyloggers.

managed services

AWS services for which AWS operates the infrastructure layer, the operating system, and platforms, and you access the endpoints to store and retrieve data. Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB are examples of managed services. These are also known as *abstracted services*.

manufacturing execution system (MES)

A software system for tracking, monitoring, documenting, and controlling production processes that convert raw materials to finished products on the shop floor.

MAP

See Migration Acceleration Program.

mechanism

A complete process in which you create a tool, drive adoption of the tool, and then inspect the results in order to make adjustments. A mechanism is a cycle that reinforces and improves itself as it operates. For more information, see Building mechanisms in the AWS Well-Architected Framework.

member account

All AWS accounts other than the management account that are part of an organization in AWS Organizations. An account can be a member of only one organization at a time.

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

A lightweight, machine-to-machine (M2M) communication protocol, based on the publish/subscribe pattern, for resource-constrained IoT devices.

microservice

A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed,

and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

Migration Acceleration Program (MAP)

An AWS program that provides consulting support, training, and services to help organizations build a strong operational foundation for moving to the cloud, and to help offset the initial cost of migrations. MAP includes a migration methodology for executing legacy migrations in a methodical way and a set of tools to automate and accelerate common migration scenarios.

migration at scale

The process of moving the majority of the application portfolio to the cloud in waves, with more applications moved at a faster rate in each wave. This phase uses the best practices and lessons learned from the earlier phases to implement a *migration factory* of teams, tools, and processes to streamline the migration of workloads through automation and agile delivery. This is the third phase of the AWS migration strategy.

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners, migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the Cloud Migration Factory guide in this content set.

migration metadata

The information about the application and server that is needed to complete the migration. Each migration pattern requires a different set of migration metadata. Examples of migration metadata include the target subnet, security group, and AWS account.

migration pattern

A repeatable migration task that details the migration strategy, the migration destination, and the migration application or service used. Example: Rehost migration to Amazon EC2 with AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO

comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

The process of gaining insights about an organization's cloud readiness status, identifying strengths and weaknesses, and building an action plan to close identified gaps, using the AWS CAF. For more information, see the migration readiness guide. MRA is the first phase of the AWS migration strategy.

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

ML

See machine learning.

modernization

Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can

use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

A process that helps generate predictions for multiple classes (predicting one of more than two outcomes). For example, an ML model might ask "Is this product a book, car, or phone?" or "Which product category is most interesting to this customer?"

mutable infrastructure

A model that updates and modifies the existing infrastructure for production workloads. For improved consistency, reliability, and predictability, the AWS Well-Architected Framework recommends the use of immutable infrastructure as a best practice.

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

A migration method in which the source workload is taken down during the migration process. This method involves extended downtime and is typically used for small, non-critical workloads.

OI

See operations integration.

OLA

See operational-level agreement.

online migration

A migration method in which the source workload is copied to the target system without being taken offline. Applications that are connected to the workload can continue to function during the migration. This method involves zero to minimal downtime and is typically used for critical production workloads.

OPC-UA

See Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

A machine-to-machine (M2M) communication protocol for industrial automation. OPC-UA provides an interoperability standard with data encryption, authentication, and authorization schemes.

operational-level agreement (OLA)

An agreement that clarifies what functional IT groups promise to deliver to each other, to support a service-level agreement (SLA).

operational readiness review (ORR)

A checklist of questions and associated best practices that help you understand, evaluate, prevent, or reduce the scope of incidents and possible failures. For more information, see Operational Readiness Reviews (ORR) in the AWS Well-Architected Framework.

operational technology (OT)

Hardware and software systems that work with the physical environment to control industrial operations, equipment, and infrastructure. In manufacturing, the integration of OT and information technology (IT) systems is a key focus for Industry 4.0 transformations.

operations integration (OI)

The process of modernizing operations in the cloud, which involves readiness planning, automation, and integration. For more information, see the operations integration guide.

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the

organization and tracks the activity in each account. For more information, see Creating a trail for an organization in the CloudTrail documentation.

organizational change management (OCM)

A framework for managing major, disruptive business transformations from a people, culture, and leadership perspective. OCM helps organizations prepare for, and transition to, new systems and strategies by accelerating change adoption, addressing transitional issues, and driving cultural and organizational changes. In the AWS migration strategy, this framework is called *people acceleration*, because of the speed of change required in cloud adoption projects. For more information, see the OCM guide.

origin access control (OAC)

In CloudFront, an enhanced option for restricting access to secure your Amazon Simple Storage Service (Amazon S3) content. OAC supports all S3 buckets in all AWS Regions, server-side encryption with AWS KMS (SSE-KMS), and dynamic PUT and DELETE requests to the S3 bucket.

origin access identity (OAI)

In CloudFront, an option for restricting access to secure your Amazon S3 content. When you use OAI, CloudFront creates a principal that Amazon S3 can authenticate with. Authenticated principals can access content in an S3 bucket only through a specific CloudFront distribution. See also OAC, which provides more granular and enhanced access control.

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

In an AWS multi-account architecture, a VPC that handles network connections that are initiated from within an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

P

permissions boundary

An IAM management policy that is attached to IAM principals to set the maximum permissions that the user or role can have. For more information, see Permissions boundaries in the IAM documentation.

personally identifiable information (PII)

Information that, when viewed directly or paired with other related data, can be used to reasonably infer the identity of an individual. Examples of PII include names, addresses, and contact information.

PII

See personally identifiable information.

playbook

A set of predefined steps that capture the work associated with migrations, such as delivering core operations functions in the cloud. A playbook can take the form of scripts, automated runbooks, or a summary of processes or steps required to operate your modernized environment.

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

An object that can define permissions (see identity-based policy), specify access conditions (see resource-based policy), or define the maximum permissions for all accounts in an organization in AWS Organizations (see service control policy).

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more

easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements.

portfolio assessment

A process of discovering, analyzing, and prioritizing the application portfolio in order to plan the migration. For more information, see Evaluating migration readiness.

predicate

A query condition that returns `true` or `false`, commonly located in a `WHERE` clause.

predicate pushdown

A database query optimization technique that filters the data in the query before transfer. This reduces the amount of data that must be retrieved and processed from the relational database, and it improves query performance.

preventative control

A security control that is designed to prevent an event from occurring. These controls are a first line of defense to help prevent unauthorized access or unwanted changes to your network. For more information, see Preventative controls in *Implementing security controls on AWS*.

principal

An entity in AWS that can perform actions and access resources. This entity is typically a root user for an AWS account, an IAM role, or a user. For more information, see *Principal* in Roles terms and concepts in the IAM documentation.

privacy by design

A system engineering approach that takes privacy into account through the whole development process.

private hosted zones

A container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs. For more information, see Working with private hosted zones in the Route 53 documentation.

proactive control

A security control designed to prevent the deployment of noncompliant resources. These controls scan resources before they are provisioned. If the resource is not compliant with the control, then it isn't provisioned. For more information, see the Controls reference guide in the

AWS Control Tower documentation and see Proactive controls in *Implementing security controls on AWS*.

product lifecycle management (PLM)

The management of data and processes for a product throughout its entire lifecycle, from design, development, and launch, through growth and maturity, to decline and removal.

production environment

See environment.

programmable logic controller (PLC)

In manufacturing, a highly reliable, adaptable computer that monitors machines and automates manufacturing processes.

prompt chaining

Using the output of one LLM prompt as the input for the next prompt to generate better responses. This technique is used to break down a complex task into subtasks, or to iteratively refine or expand a preliminary response. It helps improve the accuracy and relevance of a model's responses and allows for more granular, personalized results.

pseudonymization

The process of replacing personal identifiers in a dataset with placeholder values. Pseudonymization can help protect personal privacy. Pseudonymized data is still considered to be personal data.

publish/subscribe (pub/sub)

A pattern that enables asynchronous communications among microservices to improve scalability and responsiveness. For example, in a microservices-based MES, a microservice can publish event messages to a channel that other microservices can subscribe to. The system can add new microservices without changing the publishing service.

Q

query plan

A series of steps, like instructions, that are used to access the data in a SQL relational database system.

query plan regression

When a database service optimizer chooses a less optimal plan than it did before a given change to the database environment. This can be caused by changes to statistics, constraints, environment settings, query parameter bindings, and updates to the database engine.

R

RACI matrix

See responsible, accountable, consulted, informed (RACI).

RAG

See Retrieval Augmented Generation.

ransomware

A malicious software that is designed to block access to a computer system or data until a payment is made.

RASCI matrix

See responsible, accountable, consulted, informed (RACI).

RCAC

See row and column access control.

read replica

A copy of a database that's used for read-only purposes. You can route queries to the read replica to reduce the load on your primary database.

re-architect

See 7 Rs.

recovery point objective (RPO)

The maximum acceptable amount of time since the last data recovery point. This determines what is considered an acceptable loss of data between the last recovery point and the interruption of service.

recovery time objective (RTO)

The maximum acceptable delay between the interruption of service and restoration of service.

refactor

See 7 Rs.

Region

A collection of AWS resources in a geographic area. Each AWS Region is isolated and independent of the others to provide fault tolerance, stability, and resilience. For more information, see Specify which AWS Regions your account can use.

regression

An ML technique that predicts a numeric value. For example, to solve the problem of "What price will this house sell for?" an ML model could use a linear regression model to predict a house's sale price based on known facts about the house (for example, the square footage).

rehost

See 7 Rs.

release

In a deployment process, the act of promoting changes to a production environment.

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

An application's ability to resist or recover from disruptions. High availability and disaster recovery are common considerations when planning for resiliency in the AWS Cloud. For more information, see AWS Cloud Resilience.

resource-based policy

A policy attached to a resource, such as an Amazon S3 bucket, an endpoint, or an encryption key. This type of policy specifies which principals are allowed access, supported actions, and any other conditions that must be met.

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines the roles and responsibilities for all parties involved in migration activities and cloud operations. The matrix name is derived from the responsibility types defined in the matrix: responsible (R), accountable (A), consulted (C), and informed (I). The support (S) type is optional. If you include support, the matrix is called a *RASCI matrix*, and if you exclude it, it's called a *RACI matrix*.

responsive control

A security control that is designed to drive remediation of adverse events or deviations from your security baseline. For more information, see Responsive controls in *Implementing security controls on AWS*.

retain

See 7 Rs.

retire

See 7 Rs.

Retrieval Augmented Generation (RAG)

A generative AI technology in which an LLM references an authoritative data source that is outside of its training data sources before generating a response. For example, a RAG model might perform a semantic search of an organization's knowledge base or custom data. For more information, see What is RAG.

rotation

The process of periodically updating a secret to make it more difficult for an attacker to access the credentials.

row and column access control (RCAC)

The use of basic, flexible SQL expressions that have defined access rules. RCAC consists of row permissions and column masks.

RPO

See recovery point objective.

RTO

See recovery time objective.

runbook

A set of manual or automated procedures required to perform a specific task. These are typically built to streamline repetitive operations or procedures with high error rates.

S

SAML 2.0

An open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API operations without you having to create user in IAM for everyone in your organization. For more information about SAML 2.0-based federation, see About SAML 2.0-based federation in the IAM documentation.

SCADA

See supervisory control and data acquisition.

SCP

See service control policy.

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user credentials, that you store in encrypted form. It consists of the secret value and its metadata. The secret value can be binary, a single string, or multiple strings. For more information, see What's in a Secrets Manager secret? in the Secrets Manager documentation.

security by design

A system engineering approach that takes security into account through the whole development process.

security control

A technical or administrative guardrail that prevents, detects, or reduces the ability of a threat actor to exploit a security vulnerability. There are four primary types of security controls: preventative, detective, responsive, and proactive.

security hardening

The process of reducing the attack surface to make it more resistant to attacks. This can include actions such as removing resources that are no longer needed, implementing the security best practice of granting least privilege, or deactivating unnecessary features in configuration files.

security information and event management (SIEM) system

Tools and services that combine security information management (SIM) and security event management (SEM) systems. A SIEM system collects, monitors, and analyzes data from servers, networks, devices, and other sources to detect threats and security breaches, and to generate alerts.

security response automation

A predefined and programmed action that is designed to automatically respond to or remediate a security event. These automations serve as detective or responsive security controls that help you implement AWS security best practices. Examples of automated response actions include modifying a VPC security group, patching an Amazon EC2 instance, or rotating credentials.

server-side encryption

Encryption of data at its destination, by the AWS service that receives it.

service control policy (SCP)

A policy that provides centralized control over permissions for all accounts in an organization in AWS Organizations. SCPs define guardrails or set limits on actions that an administrator can delegate to users or roles. You can use SCPs as allow lists or deny lists, to specify which services or actions are permitted or prohibited. For more information, see Service control policies in the AWS Organizations documentation.

service endpoint

The URL of the entry point for an AWS service. You can use the endpoint to connect programmatically to the target service. For more information, see AWS service endpoints in *AWS General Reference*.

service-level agreement (SLA)

An agreement that clarifies what an IT team promises to deliver to their customers, such as service uptime and performance.

service-level indicator (SLI)

A measurement of a performance aspect of a service, such as its error rate, availability, or throughput.

service-level objective (SLO)

A target metric that represents the health of a service, as measured by a service-level indicator.

shared responsibility model

A model describing the responsibility you share with AWS for cloud security and compliance. AWS is responsible for security *of* the cloud, whereas you are responsible for security *in* the cloud. For more information, see Shared responsibility model.

SIEM

See security information and event management system.

single point of failure (SPOF)

A failure in a single, critical component of an application that can disrupt the system.

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your organization's capabilities and services, improves developer productivity, and supports rapid

innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

SPOF

See single point of failure.

star schema

A database organizational structure that uses one large fact table to store transactional or measured data and uses one or more smaller dimensional tables to store data attributes. This structure is designed for use in a data warehouse or for business intelligence purposes.

strangler fig pattern

An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was introduced by Martin Fowler as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

subnet

A range of IP addresses in your VPC. A subnet must reside in a single Availability Zone.

supervisory control and data acquisition (SCADA)

In manufacturing, a system that uses hardware and software to monitor physical assets and production operations.

symmetric encryption

An encryption algorithm that uses the same key to encrypt and decrypt the data.

synthetic testing

Testing a system in a way that simulates user interactions to detect potential issues or to monitor performance. You can use Amazon CloudWatch Synthetics to create these tests.

system prompt

A technique for providing context, instructions, or guidelines to an LLM to direct its behavior. System prompts help set context and establish rules for interactions with users.

T

tags

Key-value pairs that act as metadata for organizing your AWS resources. Tags can help you manage, identify, organize, search for, and filter resources. For more information, see Tagging your AWS resources.

target variable

The value that you are trying to predict in supervised ML. This is also referred to as an *outcome variable*. For example, in a manufacturing setting the target variable could be a product defect.

task list

A tool that is used to track progress through a runbook. A task list contains an overview of the runbook and a list of general tasks to be completed. For each general task, it includes the estimated amount of time required, the owner, and the progress.

test environment

See environment.

training

To provide data for your ML model to learn from. The training data must contain the correct answer. The learning algorithm finds patterns in the training data that map the input data attributes to the target (the answer that you want to predict). It outputs an ML model that captures these patterns. You can then use the ML model to make predictions on new data for which you don't know the target.

transit gateway

A network transit hub that you can use to interconnect your VPCs and on-premises networks. For more information, see What is a transit gateway in the AWS Transit Gateway documentation.

trunk-based workflow

An approach in which developers build and test features locally in a feature branch and then merge those changes into the main branch. The main branch is then built to the development, preproduction, and production environments, sequentially.

trusted access

Granting permissions to a service that you specify to perform tasks in your organization in AWS Organizations and in its accounts on your behalf. The trusted service creates a service-linked role in each account, when that role is needed, to perform management tasks for you. For more information, see Using AWS Organizations with other AWS services in the AWS Organizations documentation.

tuning

To change aspects of your training process to improve the ML model's accuracy. For example, you can train the ML model by generating a labeling set, adding labels, and then repeating these steps several times under different settings to optimize the model.

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development.

U

uncertainty

A concept that refers to imprecise, incomplete, or unknown information that can undermine the reliability of predictive ML models. There are two types of uncertainty: *Epistemic uncertainty* is caused by limited, incomplete data, whereas *aleatoric uncertainty* is caused by the noise and randomness inherent in the data.

undifferentiated tasks

Also known as *heavy lifting*, work that is necessary to create and operate an application but that doesn't provide direct value to the end user or provide competitive advantage. Examples of undifferentiated tasks include procurement, maintenance, and capacity planning.

upper environments

See environment.

V

vacuuming

A database maintenance operation that involves cleaning up after incremental updates to reclaim storage and improve performance.

version control

Processes and tools that track changes, such as changes to source code in a repository.

VPC peering

A connection between two VPCs that allows you to route traffic by using private IP addresses. For more information, see What is VPC peering in the Amazon VPC documentation.

vulnerability

A software or hardware flaw that compromises the security of the system.

W

warm cache

A buffer cache that contains current, relevant data that is frequently accessed. The database instance can read from the buffer cache, which is faster than reading from the main memory or disk.

warm data

Data that is infrequently accessed. When querying this kind of data, moderately slow queries are typically acceptable.

window function

A SQL function that performs a calculation on a group of rows that relate in some way to the current record. Window functions are useful for processing tasks, such as calculating a moving average or accessing the value of rows based on the relative position of the current row.

workload

A collection of resources and code that delivers business value, such as a customer-facing application or backend process.

workstream

Functional groups in a migration project that are responsible for a specific set of tasks. Each workstream is independent but supports the other workstreams in the project. For example, the portfolio workstream is responsible for prioritizing applications, wave planning, and collecting migration metadata. The portfolio workstream delivers these assets to the migration workstream, which then migrates the servers and applications.

WORM

See write once, read many.

WQF

See AWS Workload Qualification Framework.

write once, read many (WORM)

A storage model that writes data a single time and prevents the data from being deleted or modified. Authorized users can read the data as many times as needed, but they cannot change it. This data storage infrastructure is considered immutable.

Z

zero-day exploit

An attack, typically malware, that takes advantage of a zero-day vulnerability.

zero-day vulnerability

An unmitigated flaw or vulnerability in a production system. Threat actors can use this type of vulnerability to attack the system. Developers frequently become aware of the vulnerability as a result of the attack.

zero-shot prompting

Providing an LLM with instructions for performing a task but no examples (*shots*) that can help guide it. The LLM must use its pre-trained knowledge to handle the task. The effectiveness of zero-shot prompting depends on the complexity of the task and the quality of the prompt. See also few-shot prompting.

zombie application

An application that has an average CPU and memory usage below 5 percent. In a migration project, it is common to retire these applications.