



Accelerating software development lifecycles on AWS with generative AI

AWS Prescriptive Guidance



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

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Introduction	1
Objectives	1
Intended audience	2
Development experience	3
Using generative AI	4
5-I framework	5
Framework overview	6
Integrating with the software development lifecycle	8
Foundational capabilities	9
Project management	14
Requirement management	17
Architecture and design	18
Collaboration	19
DevSecOps	20
Operation and maintenance	28
AI assistants	30
Analytics and insights	32
Knowledge management	34
Extensibility	35
Best practices	37
Integrated toolchain	37
DevSecOps pipeline	38
Collaborative tools and practices	38
Task automation	38
Review and iteration	39
Project management practices	39
Knowledge management	40
Extensibility and customization	40
Optimization	40
Data-driven insights	41
Platform-based approach	41
Measuring success	42
Deployment velocity	43
Code quality	43

Operational efficiency	44
Team productivity and satisfaction	44
Business impact	45
Conclusion	46
Resources	46
AWS documentation	46
AWS blog posts and tutorials	47
Document history	48

Accelerating software development lifecycles on AWS with generative AI

Chetan Makvana, Amazon Web Services

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.

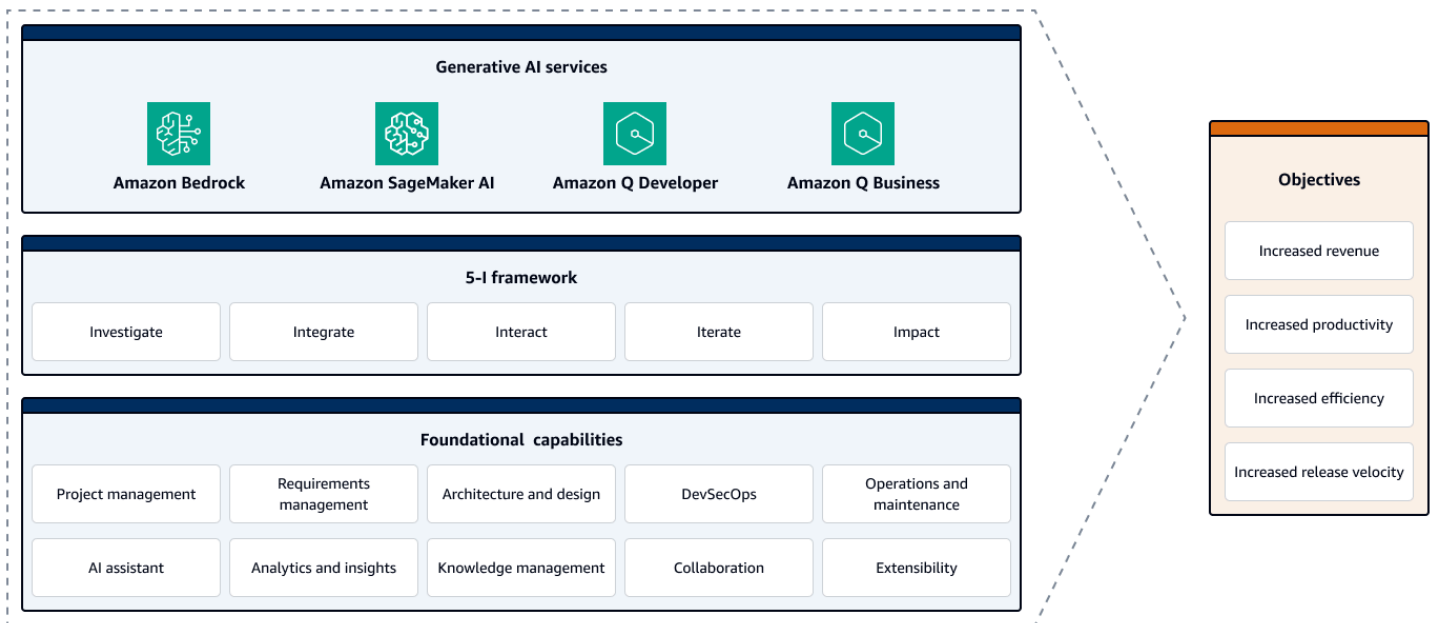
At its core, a generative AI-powered development experience acts as an intelligent collaborator for your entire software development team, including product managers, designers, solutions architects, developers, testers, and operations personnel. It provides context-aware assistance, generates artifacts (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.

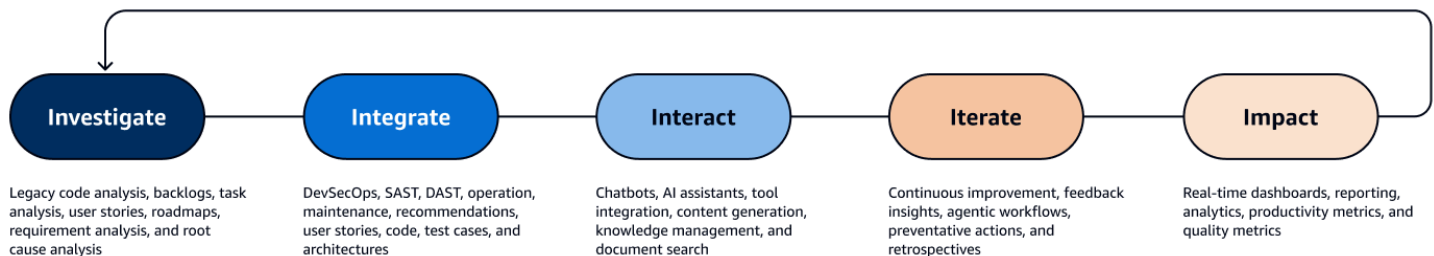
This section contains the following topics:

- [Framework overview](#)
- [Integrating with the software development lifecycle](#)

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
Project management:	Requirements and planning	None	None	None	Requirements and planning

**t: Reporting
and analytics**

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
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
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 maintenan ce: Incident management	None	None	None	Operation and maintenance	None
Operation and maintenan ce: Code upgrades	None	Operation and maintenance	None	None	None
Operation and maintenan ce: Code optimization	Operation and maintenance	Operation and maintenance	None	None	None
Operation and maintenan ce: Technical debt management	None	Operation and maintenance	Operation and maintenance	None	None
Operation and maintenan ce: Change management	None	Implement ation, 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
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: Distribute 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
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	Product manager
Product roadmap management: Update the roadmap based on changes in project priorities or timelines	Product manager
Product roadmap management: Share the roadmap with stakeholders to provide visibility 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
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 , share knowledge, and troubleshoot issues collaboratively	Development team
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
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: Automatically 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

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

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 production 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
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 immediately 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
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-based 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 applications 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 vulnerabilities	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
Application performance monitoring: Continuously monitor application performance 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 distributed system to identify performance bottlenecks 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
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 dashboards 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 infrastructure	Software developer
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
Code upgrades: Upgrade code dependencies 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 optimization 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
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 developer
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
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 opportunities 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
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
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 development environment	DevOps engineer
Create custom automation workflows that are tailored to team's unique development process	DevOps engineer
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](#)

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.

DevSecOps pipeline

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.

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.

Task automation

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.

Review and iteration

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.

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.

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.

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.

Optimization

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.

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.

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](#)
- [Code quality](#)
- [Operational efficiency](#)
- [Team productivity and satisfaction](#)
- [Business impact](#)

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 percentage for each task
Satisfaction score	Conduct regular surveys to gauge improvement 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.

Change	Description	Date
Initial publication	Not applicable	April 18, 2025