Introduction to Developing on AWS

01 MUST KNOW

if you remember five things, remember these

01 **Develop is the focus:** The **SDLC** (systems development lifecycle) runs **plan** → **define** → **design** → **develop** → **deploy** → **maintain**; this course centers on **Develop** and briefly touches **Deploy**. Development itself breaks into five phases — **Code, Build, Test, Deploy, Maintain** — where **Code, Build, and Test** fall under the Develop stage. The goal: deliver software *quickly, securely, and reliably* with **zero tolerance for outages**.

02 **Waterfall vs. agile: Waterfall** is *sequential* — each phase must finish before the next begins, so customer feedback can be months away. **Agile** is *iterative*: teams ship fast in short **sprints** (typically **1–4 weeks**) to collect feedback each cycle (MVP, release candidate, velocity). Developing on AWS makes **frequently changing requirements** easier to absorb.

03 **Four steps to start:** Before you build on AWS: **(1)** create an **AWS account**, **(2)** set up permissions with **IAM**, **(3)** install the **development environment** (your language plus an IDE and AWS IDE Toolkit), and **(4)** interact with AWS services using the **AWS SDKs** and the **AWS CLI**.

04 **One API, many front doors:** Every AWS service is managed through a common **REST-like API**, and each service has its own **endpoint** (a URL entry point). You reach it four *interchangeable* ways: **directly**, the **Management Console**, the **AWS CLI**, or an **AWS SDK**. A CLI call begins with `aws`, then the **service**, the **operation** subcommand, any **parameters** (argument names prefixed with `--`), and optional **options** like `--query`.

05 **Amazon Q Developer:** A **generative AI-powered coding assistant** for developers and IT pros — even those without deep experience. It chats about code, gives in-line completions, generates new code, and **scans code for security vulnerabilities**, supporting **every phase of the SDLC**. It is secure and private by design.

02 KEY TERMS

TERM	DEFINITION
Systems development lifecycle (SDLC)	Conceptual project-management model of the stages of an information-system project: plan, define, design, develop, deploy, maintain
Waterfall methodology	Classic *sequential* approach; each phase must be completed before the next begins, so feedback can be slow
Agile methodology	Iterative framework that pushes work to customers quickly to collect feedback and improve each iteration
Sprint	A short, fixed iteration in agile development, typically lasting 1–4 weeks
IAM	AWS Identity and Access Management — controls who can use resources (authentication) and what they can do (authorization) via users, groups, roles, and policies
AWS CLI	Open source command line tool for interacting with AWS services from a terminal on Windows, macOS, or Linux/Unix
AWS CloudShell	Browser-based, pre-authenticated CLI in the console; pre-installed with the AWS CLI, SDKs, and tools at no cost, with 1 GB storage per Region
AWS SDK	Language-specific packages that let application code call AWS services and resources
Service client API	Low-level SDK API with one method per service operation and objects for the request and result — full control
Resource API	Higher-level, object-oriented SDK API with one class per conceptual resource (e.g., Boto3); not offered by every SDK

TERM	DEFINITION
Endpoint	A URL that is the entry point for a web service; the SDK sends its HTTP(S) request to the service endpoint
Amazon Q Developer	Generative AI coding assistant that helps write, understand, build, extend, and operate AWS applications and scans code for security vulnerabilities

03 FOUR STEPS TO GET STARTED DEVELOPING ON AWS

set up before you write code

`01` Create an **AWS account** — sign up at `aws.amazon.com` → `02` Set up permissions with **IAM** — users, groups, roles, policies → `03` Install the **development environment** — language + IDE and AWS IDE Toolkit → `04` Interact with **AWS services** — AWS SDKs and the AWS CLI

04 WATERFALL VS. AGILE

the two most common SDLC methodologies

WATERFALL (TRADITIONAL)

sequential — finish a phase, then start the next

- Classic, sequential development method
- Each phase has distinct goals completed before the next begins
- Customer feedback may not arrive for months — often not until commercialization
- Best when requirements are stable and well understood

AGILE

iterative — ship fast, learn, improve

- Fast-paced, iterative framework; push work to customers quickly
- Work in **sprints** — short cycles, typically **1–4 weeks**
- Concepts: MVP, release candidate, velocity
- Methods: Scrum, Crystal, DSDM; requirements can change often

05 SERVICE CLIENT API VS. RESOURCE API

the two API levels an SDK provides

SERVICE CLIENT API

low-level · full control

- One method per service operation; objects for request and result
- A direct, 1:1 mapping of the service API
- Tight control over requests, behavior, and performance
- Python: `boto3.client('s3')` then `list_objects_v2(...)` returns a `dict`

RESOURCE API

higher-level · object-oriented

- One class per conceptual resource (an S3 bucket, an object)
- A higher-level abstraction over the low-level client calls
- Access through objects and collections; methods and attributes
- Python: `boto3.resource('s3')` then `bucket.objects.all()` — Boto3 has it; not every SDK does

06 AMAZON Q DEVELOPER ACROSS THE SDLC

one assistant, every phase

SDLC PHASE	WHAT AMAZON Q DEVELOPER DOES
Plan	Ask questions and get referenceable, contextual guidance; explain code with conversational coding
Create	In-line code recommendations for multiple languages; implement features from comments or prompts; chat in the IDE
Test & secure	Generate unit tests; scan project code for security vulnerabilities and get remediation suggestions
Operate	Troubleshoot errors; troubleshoot network connectivity with VPC Reachability Analyzer
Maintain & modernize	Upgrade and modernize code with the Amazon Q Developer Agent for code transformation

07 EXAM TRAPS

where the knowledge check gets you

- × "Just use the root account for everything." — The **root user** has complete access to every service and resource; best practice is to **not use it unless required** and instead create an **IAM user** with its own access credentials.
- × "Waterfall is iterative like agile." — Backwards. **Waterfall is sequential** (finish one phase before the next); **agile** is the iterative one, shipping in short **sprints** to gather feedback each cycle.
- × "The service client API and the resource API are two different SDKs." — They are **two API levels inside one SDK**: the client API is a low-level 1:1 mapping; the resource API is a higher-level object-oriented abstraction — and not every SDK provides one (**Boto3** does).
- × "A 400-series error means retry." — A **400 (client) error** must be **handled in your application** (e.g., create the missing S3 bucket); a **500 (server) error** is the one you **retry**, and SDKs retry automatically with **exponential backoff**.
- × "You always create one service client per Region." — Some SDKs (**Java, .NET**) need a separate client instance per Region; others (**Boto3**) can use the **default Region** from `~/.aws/config`.
- × "Amazon Q Developer only writes code." — It supports the **whole SDLC** — Plan, Create, Test & secure, Operate, and Maintain & modernize — including generating unit tests and **scanning code for security vulnerabilities**.
- × "AWS CloudShell costs extra and needs key files." — CloudShell is **no cost** and **pre-authenticated** (it inherits your console credentials, so there are no key files to manage), with the CLI, SDKs, and tools pre-installed plus **1 GB of storage per Region**.
- × "The Management Console always has every AWS feature first." — Occasionally a **new feature is not yet available in the console** when it launches; the CLI, SDKs, and direct API are equally valid, interchangeable ways to reach the same service.

08 SELF-QUIZ

answers are upside-down below

Q1 Which phase of the SDLC does this course focus on?

Q2 Which SDLC methodology develops software sequentially, requiring each phase to complete before the next begins?

Q3 Under agile, what is the short, typically 1–4 week iteration called?

Q4 Name the four steps to get started developing on AWS, in order.

Q5 Which AWS service controls who can access resources (authentication) and what they can do (authorization)?

Q6 What are the two levels of API that the AWS SDKs provide for calling AWS services?

Q7 In an AWS CLI command, what character sequence precedes each parameter (argument) name?

Q8 An AWS service returns a 500-series error. What should your application do?

Q9 Which browser-based tool provides a pre-authenticated CLI with the AWS CLI and SDKs pre-installed at no cost?

Q10 Which generative AI-powered coding assistant helps developers across the SDLC and scans code for security vulnerabilities?

ANSWER KEY (rotate the page)

Q1 the Develop phase (it also briefly touches Deploy) **Q2** waterfall (traditional) **Q3** a sprint **Q4** create an AWS account → set up permissions with IAM → install the development environment → interact with AWS services **Q5** IAM (Identity and Access Management) **Q6** the service client API (low-level) and the resource API (higher-level) **Q7** two dashes (--) **Q8** retry the operation (SDKs retry automatically with exponential backoff) **Q9** AWS CloudShell **Q10** Amazon Q Developer