AWS Cloud Institute: Developer Intermediate 2 Syllabus

Course overview

In this course, you are challenged to design, build, test, and deploy microservices by emulating the activities of AWS developers. You are exposed to various AWS services and are required to make design choices, write code, troubleshoot issues, and iterate on your solution to meet new business requirements. The course aims to cultivate a developer mindset, enabling you to think critically, leverage relevant tools, and gain practical experience in implementing cloud-based applications.

Instructor-led training (ILT) sessions

Attendance at ILT sessions is strongly encouraged, but attendance does not count toward course completion. You can use the ILT sessions to explore concepts presented in the online learning content in greater depth, connect with AWS instructors and ACI learners, and prepare for the weekly final assessment. All ILT sessions, except for office hours, are recorded and will be available on-demand for you to watch when you have time. Multiple live ILT sessions are offered each day to accommodate a range of schedules.

Course completion requirements

At the end of each module, you will complete a multiple-choice final assessment. You have unlimited opportunities to achieve a passing score on each final assessment. You must complete the final assessments and the cumulative course assessment with a score of 80% or higher to receive credit for the course. ILT attendance or watching ILT recordings is strongly encouraged but is not required for course completion.

Course content updates

ACI courses are developed within a dynamic, fast-paced technical environment where products and services continually change. Therefore, this syllabus serves as a guide, but the actual content delivered in the course may be updated to stay current.

Capstone Project: Building and Deploying an Appointments Scheduler Application

Goal

In this capstone project, you will design and develop a working application to demonstrate your ability to use AWS services and Python code to create a cloud-based solution to an employer-informed problem scenario.

Hands-on labs

- **Lab: Building and Deploying an Appointments Scheduler App - Lab 01 (60 min)**
  In this lab, you will clone the application code from a Git repository, run the application in an integrated development environment (IDE), and improve the percentage of code coverage by adding a new unit test.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 02 (60 min)**
  In this lab, you will update application code to add a new feature and verify it by writing a new unit test.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 03 (60 min)**
  In this lab, an Amazon DynamoDB table has been created, and you will update the Python code to add a new feature in the app that enables making calls to a DynamoDB table. Additionally, you will write a unit test mocking the API calls to DynamoDB table.

- **Lab: Building and Deploying an Appointments Scheduler App - Lab 04 (60 min**)
  In this lab, you will build a continuous integration pipeline that automates testing the application code when a developer commits changes.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 05 (90 min)**
  In this lab, you will configure the application to work with an Amazon Relational Database Service (Amazon RDS) instance instead of using SQLite. You will build a Docker container image of your application, test the image locally in your integrated development environment (IDE), and then push the image to an Amazon Elastic Container Registry (Amazon ECR) repository.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 06 (90 min)**
  In this lab, you will use AWS CodePipeline and AWS CodeBuild to add a new stage to the continuous integration pipeline and automate building the container image and pushing it to Amazon Elastic Container Repository (Amazon ECR).
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 07 (60 min)**
  In this lab, you will deploy the application on a Kubernetes cluster, which is managed by Amazon Elastic Kubernetes Service (Amazon EKS), and verify it.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 08 (60 min)**
  In this lab, you will learn how to troubleshoot application issues by viewing the logs using the *kubectl* utility. You will also learn how to perform application rollbacks.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 09 (60 min)**
  In this lab, you will update the Kubernetes deployment to replace the Classic Load Balancer with an Application Load Balancer.
- **Lab: Building and Deploying an Appointments Scheduler App - Lab 10 (60 min)**
  In this lab, you will add a new stage to the AWS CodePipeline pipeline to automate deploying the application onto the Kubernetes cluster. You will also verify the pipeline, make a code update, and perform a rollback using the Git revert feature.

Week 1: Developing on AWS Overview

Goal

This week, you will review the development process and learn about AWS services, like Amazon Q Developer, to plan a microservices application for AnyCompany Pet Shelter. You will also develop the frontend of the AnyCompany Pet Shelter application in an integrated development environment (IDE).

Learning objectives

- Review the agile software development process
- Review essential AWS developer tools
- Identify the features of Amazon Q Developer that accelerate the development process
- Review the characteristics of a modern application architecture
- Review microservice storage, implementation, and integration patterns
- Design a microservices architecture for a new web application
- Decompose a monolithic web application into microservices
- Become familiar with popular web libraries and frameworks
- Understand how TypeScript is related to JavaScript
- Create a React application using *npm* commands and analyze key React application files and settings
- Describe the fundamental architecture of a React application
- Learn how to test a React application
- Recognize the importance of using a version control system, including how Git commands are used

Outline

1. **Developing on AWS**
    a. **Getting Started**
        i. Business Case Studies Used in the Course
        ii. Practice Environment: Frontend Web Application Development with React
    b. **Developing on AWS**
        i. Agile Software Development Process
        ii. AWS Development Tools
        iii. Knowledge Check
    c. **Designing a Microservices Application**
        i. Modern Application Architecture Review
        ii. Activity: Designing the Pet Shelter Application
        iii. Activity: Refactoring the Bicycle Parts Application
        iv. Knowledge Check
    d. **Developing Application Frontends with React**
        i. Overview of Popular Web Libraries and Frameworks
        ii. Activity: Working with React
        iii. Activity: Building a User Interface with React
        iv. Activity: Testing a React Application
        v. Importance of Using a Version Control System
        vi. Lab: Building a Frontend with a Code Framework
        vii. Knowledge Check

Hands-on labs

- **Practice Environment: Frontend Web Application Development with React (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Building a Frontend with a Code Framework (90 min)**
  Connect to the IDE, install software needed to support AWS development, and then create a web application using React, Vite, and JavaScript.

Week 2: Building the Pets Microservice

Goal

This week, you will build the first microservice for the AnyCompany Pet Shelter application, called the Pets microservice. To build this microservice, you will use AWS Serverless Application Model (AWS SAM), author Lambda functions, and update React code to pull a list of available pets at the shelter.

Learning objectives

- Review the serverless architecture of a microservice
- Practice the fundamentals of routing and request handling with Amazon API Gateway
- Use AWS SAM to efficiently create and deploy API Gateway and Lambda integrations
- Develop and deploy serverless functions in Python using best practices for scalability and performance
- Practice troubleshooting common errors in serverless applications

- Author a Lambda function that responds to an API Gateway Request, calls DynamoDB, and returns a response to API Gateway
- Understand how to capture errors that occur in Lambda function code and return an appropriate response for API Gateway
- Understand different methods to populate a DynamoDB table
- Learn how to manually add items to a DynamoDB table using the AWS Console
- Learn how to create and execute a Python script to populate a DynamoDB table
- Understand how to update React code to fetch data from an API
- Understand the importance of using environment variables for storing URLs
- Learn how to create and configure an Amazon Simple Storage Service (Amazon S3) bucket for storing and serving images
- Understand how to upload images to an Amazon S3 bucket and use them in a React application

Outline

1. **Building the Pets Microservice**
    a. **Getting Started**
        i. Practice Environment: Building the Pets Microservice
    b. **Reviewing Serverless Microservices with AWS SAM**
        i. Serverless Microservices with AWS SAM
        ii. The Pets Microservice Architecture
        iii. Knowledge Check
    c. **Setting Up API Gateway To Handle Requests Routing**
        i. Activity: Adding an API Gateway to the AWS SAM Template
        ii. Activity: Adding an Implicit API to the AWS SAM Template
        iii. Knowledge Check
    d. **Building and Testing the Microservice Lambda Function**
        i. Activity: Updating the Lambda Function Code to Call DynamoDB
        ii. Knowledge Check
    e. **Using DynamoDB to Store Data in a Serverless Application**
        i. Activity: Defining a DynamoDB Table using AWS SAM
        ii. Activity: Populating the DynamoDB Table
        iii. Knowledge Check
    f. **Integrating the Microservice with a React Frontend**
        i. Activity: Updating the React Code to Fetch data from the API
        ii. Activity: Uploading Images to an Amazon S3 Bucket
        iii. Knowledge Check
    g. **Hands-On Lab Activity**
        i. Lab: Creating a Get Products Microservice

Hands-on labs

- **Practice Environment: Building the Pets Microservice (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Creating a Get Products Microservice (150 min)**
  You use an AWS supported integrated development environment (IDE) and AWS Serverless Application Model (AWS SAM) for defining the infrastructure as code (IaC) serverless application microservice for retrieving product information. You will also use Amazon DynamoDB as your NoSQL data storage solution, AWS Lambda to run code on demand, and Amazon API Gateway service to create API endpoints. Finally, rather than code Python from scratch, in this lab, you will prompt Amazon Bedrock to author code that you can use as part of your microservices-based serverless solution.

Week 3: Building the Adoptions Microservice Part 1

Goal

In this module, you will build the Adoptions microservice, which allows AnyCompany Pet Shelter employees to pull information about applications for adoption in the pet shelter application. You will build on your knowledge of AWS SAM, Amazon DynamoDB, and AWS Lambda to pull details about adoptions for this microservice.

Learning objectives

- Use Amazon Q Developer to speed up the application development process
- Explain the benefits of adding application features through a second microservice
- Describe how to define and deploy resources for a second microservice using AWS Serverless Application Model (AWS SAM)
- Use nested attributes in Amazon DynamoDB to efficiently model and manage one-to-many relationships
- Analyze and resolve application errors using error messages and application logs
- Evaluate the pros and cons of single Lambda and multi Lambda approaches to building microservice features
- Explain how to use an AWS Lambda function's *event* parameter to retrieve path variables
- Explain how to retrieve data from Amazon DynamoDB for a single item based on its ID
- Explain how to handle different response scenarios in an AWS Lambda function to return appropriate HTTP status codes and response bodies

Outline

1. **Building the Adoptions Microservice Part 1**
   a. **Getting Started**
      i. Practice Environment: Building and Deploying the Adoptions Microservice
   b. **Getting All Adoptions**
      i. Using Amazon Q Developer
      ii. Activity: Creating a New Lambda Resource and API Endpoint
      iii. Activity: Testing Your API Gateway Endpoint
      iv. Adoption Application Data Considerations
      v. Activity: Setting up Your DynamoDB Adoptions Table
      vi. Activity: Updating Your Lambda Function and Client Code
      vii. Application Checkpoint
      viii. Knowledge Check
   c. **Getting Adoption Details**

i. Activity: Updating AWS SAM Template and Placeholder Lambda
ii. Updating Your Client Application
iii. Knowledge Check

d. Hands-On Lab

i. Lab: Creating a Get Orders and Order Details Microservice

Hands-on labs

- **Practice Environment: Building and Deploying the Adoptions Microservice (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Creating a Get Orders and Order Details Microservice (60 min)**
  In this lab, you will build a microservice to retrieve order information using an IDE as the development environment and AWS Serverless Application Model (AWS SAM) for defining the Infrastructure as Code (IaC).

Week 4: Building the Adoptions Microservice Part 2

Goal

This week, you will build a microservice that allows a prospective pet adopter to apply online for pets at AnyCompany Pet Shelter. You will build on the previous module by authoring code using Amazon Q Developer and troubleshooting common errors related to posting adoption application data to the shelter's DynamoDB database.

Learning objectives

- Add the HTTP POST method to enter data into an application
- Test a POST request to confirm its functionality
- Use HTTP status codes to monitor POST requests
- Write Lambda code to handle POST requests
- Use AWS SAM to configure and deploy Lambda functions that can respond to POST requests
- Test POST requests using cURL
- Package data in the body of a POST request

Outline

1. **Building the Adoptions Microservice Part 2**
   a. Getting Started
      i. Practice Environment: Building the Adoptions Microservice Part 2
   b. Understanding How POST Requests Work
      i. Scaling the Adoption Application Process with a New Microservice
      ii. How HTTP POST Requests Work
      iii. Ways to Test POST Method
      iv. Knowledge Check
   c. Building Out a POST Request Feature Using Amazon Q Developer
      i. Activity: Configuring and Testing the Lambda Function to Handle POST Requests
      ii. Knowledge Check
   d. Hands-On Lab Activity
      i. Lab: Creating the Process Orders Microservice

Hands-on labs

- **Practice Environment: Building the Adoptions Microservice Part 2 (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Creating the Process Orders Microservice (60 min)**
  You will use Amazon DynamoDB for storing data using a POST request, AWS Lambda to run microservice code, and the Amazon API Gateway to create API endpoints. You should also be able to debug and identify cross-origin resource sharing (CORS) issues in your web application and enable CORS requests for API Gateway.

Week 5: Securing the Application

Goal

This week, you will focus on the security of the AnyCompany Pet Shelter application and create authentication and authorization microservices using Amazon Cognito.

Learning objectives

- Review application authentication and authorization concepts
- Describe the components of Amazon Cognito and recognize its customer identity and access management benefits
- Use Amazon Cognito user pools
- Describe the purpose of Amazon Cognito identity pools

Outline

1. **Securing the Application**
    a. **Getting Started**
        i. Practice Environment: Securing the Application
    b. **Authentication and Authorization with Amazon Cognito**
        i. Securing an Application
        ii. Activity: Pet Shelter – Authentication and Authorization Part 1
        iii. Access Tokens
        iv. Activity: Pet Shelter – Authentication and Authorization Part 2
        v. Activity: Pet Shelter – Authentication and Authorization Part 3
        vi. Knowledge Check
    c. **Hands-On Lab Activity**
        i. Lab: Adding Authentication and Authorization to an App

Hands-on labs

- **Practice Environment: Securing the Application (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Adding Authentication and Authorization to an App (120 min)**
  In this lab, you will use Amazon Cognito to integrate an authentication and authorization mechanism into the website.

Week 6: Adding a Reporting Microservice

Goal

This week, you will configure Amazon Simple Notification Service (SNS) and AWS Step Functions with AWS SAM to deliver email reports from AnyCompany Pet Shelter. This allows employees of the shelter to generate adoption reports from a DynamoDB table and share them via email using presigned URLs.

Learning objectives

- Describe key features of AWS Services that enable reporting functionality
- Evaluate how to configure Amazon SNS and Step Functions with AWS SAM
- Create a reporting service using DynamoDB, AWS Lambda, Step Functions, and Amazon SNS
- Troubleshoot adding a Lambda function to a state machine

Outline

1. **Adding a Reporting Microservice**
    a. **Getting Started**
        i. Practice Environment: Adding Reporting Functionality
    b. **Configuring a State Machine for Report Generation**
        i. Reporting Microservice Architecture
        ii. Lambda Functions that Generating a Report Workflow
        iii. Activity: Adding a State Machine to the AWS SAM Template
        iv. Activity: Configuring State Machine States
        v. Knowledge Check
    c. **Implementing Lambda Functions for Reporting**
        i. Activity: Adding Lambda Functions to a State Machine
        ii. Knowledge Check
    d. **Setting Up Amazon SNS for Report Delivery**
        i. Amazon SNS Key Concepts and Architectural Benefits
        ii. Activity: Creating an Amazon SNS Topic in an AWS SAM Template
        iii. Knowledge Check
    e. **Sending a Report From the Pet Shelter Application**
        i. Activity: Using API Gateway to Start a Step Functions Execution
        ii. Activity: Triggering the Reporting Workflow from the Pet Shelter Application
        iii. Knowledge Check
    f. **Hands-On Lab Activity**
        i. Lab: Emailing an Inventory Report on Demand

Hands-on labs

- **Practice Environment: Adding Reporting Functionality (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Emailing an Inventory Report on Demand (180 min)**
  In this lab, you will use AWS Step Functions to coordinate the actions needed to generate and deliver an inventory report, upon request, by an authenticated and authorized user.

Week 7: Observability and Tracing

Goal

This week, you will use Amazon CloudWatch and AWS X-Ray to monitor resources, collect metrics, and analyze the behavior of the AnyCompany Pet Shelter application.

Learning objectives

- Configure monitoring for AWS Lambda functions and API Gateway requests
- Learn how to automatically collect and analyze CloudWatch logs for AWS Lambda functions
- Learn how to create custom metrics from CloudWatch logs

Outline

1. **Observability and Tracing**
    a. **Getting Started**
        i. Practice Environment: Observability and Tracing
    b. **Monitoring And Troubleshooting Microservice Applications**
        i. Monitoring Microservice Applications
        ii. Identifying Microservice Errors
        iii. Monitoring Lambda Using CloudWatch Logs
        iv. Activity: Monitoring Lambda Using CloudWatch
        v. Monitoring Step Functions Using CloudWatch
        vi. Activity: Demonstrating Step Functions Retry Mechanisms
        vii. Knowledge Check
    c. **Using X-Ray to Monitor and Debug Microservices**
        i. AWS X-Ray
        ii. Activity: Configuring End-to-End Trace Data Using X-Ray
        iii. Activity: Using X-Ray Trace Tables for Analysis and Troubleshooting
        iv. Activity: Using X-Ray Trace Maps
        v. Knowledge Check
    d. **Hands-On Lab Activity**
        i. Lab: Instrumenting a Distributed Application to Identify Performance Bottlenecks

Hands-on labs

- **Practice Environment: Observability and Tracing (10 min)**
  As you are presented with the exercises, perform them in the practice environment, and then proceed to the next lesson. Before you start the next lesson, choose the LAUNCH button at the bottom of this lesson to give the environment time to spin up.
- **Lab: Instrumenting a Distributed Application to Identify Performance Bottlenecks (120 min)**
  In this lab, you will use AWS X-Ray to analyze the performance of a distributed application, identify and troubleshoot performance issues, and verify the resolution with tracing.

Week 8: Using AWS Amplify Gen 2

Goal

This week, you will follow along with Example Organization Pet Shelter's frontend developer as she replicates AnyCompany's pet adoption application using AWS Amplify Gen 2. You will learn how Amplify expedites the process of

building React frontends and creates backend logic for applications. You will also learn about using Amplify Auth to add authorization and authentication, backed by Amazon Cognito.

Learning objectives

- Describe the key Amplify Gen 2 features for frontend and backend app development
- Understand the Amplify Gen 2 backend capabilities
- Describe Amplify Gen 2 Data and Amplify Gen2 Storage features
- Recognize the GraphQL features provided by AppSync and how Amplify Gen 2 can use AppSync
- Practice using Amplify CLI commands and understand what they accomplish
- Install and configure Amplify Gen 2 for a frontend and add Amazon Cognito authentication to the app

Outline

1. **Using AWS Amplify Gen 2**
    a. **Getting Started**
        i. About the Demonstrations
    b. **Setting Up an Amplify Application**
        i. Amplify as a Solution
        ii. Amplify Features
        iii. Demo: Setting Up Amplify and Adding Authentication
        iv. Knowledge Check
    c. **Building an Amplify Application**
        i. Demo: Building the Data Layer and Using Amplify Storage
        ii. Demo: Creating and Viewing Pet Data
        iii. Demo: Creating and Viewing Adoption Applications
        iv. Activity: Reflection
        v. Knowledge Check

Hands-on labs

None

Week 9: AWS Amplify Hosting

Goal

This week, you will learn how to host the frontend application for Example Organization Pet Shelter using AWS Amplify Hosting. You will follow along as the frontend developer executes manual and automated deployments. Once the application is deployed, you will monitor CloudWatch metrics and learn about troubleshooting best practices.

Learning objectives

- Describe the features and benefits of AWS Amplify Hosting
- Explain the AWS Amplify Hosting deployment options
- Deploy a web application manually using AWS Amplify Hosting
- Deploy a web application automatically using AWS Amplify Hosting
- Describe best practices for AWS Amplify Hosting

Outline

1. **AWS Amplify Hosting**
    a. **Getting Started**
        i. About the Demonstrations
    b. **Deploying a Web Application to AWS Amplify Hosting**
        i. AWS Amplify Hosting Overview
        ii. Using AWS Amplify Hosting
        iii. Demonstration: Performing a Manual Deployment
        iv. Demonstration: Performing an Automated Deployment
        v. Monitoring Application Performance and Access
        vi. Troubleshooting and Best Practices
        vii. Knowledge Check

Hands-on labs

None

Week 10: Serverless in Practice: Building Real-World Solutions

Goal

This week, you will create and deploy RESTful APIs by integrating AWS services, including API Gateway and AWS Lambda. You will also build single-page applications using Amazon S3 while practicing troubleshooting techniques for serverless architectures.

Learning objectives

- Practice building and deploying RESTful APIs in AWS SimuLearn
- Deploy and connect backend and frontend services using AWS SAM, Amazon Cognito, and AWS Amplify
- Use your knowledge of Amazon S3 bucket policies to create a single-page application in SimuLearn

Outline

1. **Serverless in Practice: Building Real-World Solutions**
    a. **Getting Started**
        i. Proactive Learning Approach
    b. **Hands-On Practice**
        i. AWS SimuLearn: Deploying RESTful APIs
        ii. AWS SimuLearn: Single-Page App

Hands-on labs

- **AWS SimuLearn: Deploying RESTful APIs (60 min)**
  This hands-on lab will guide you through creating an API Gateway REST API and connecting it to a Lambda function, simulating a common real-world scenario where APIs serve as the interface between client applications and backend services.
- **AWS SimuLearn: Single-Page App (60 min)**
  In this hands-on lab, you will deploy and troubleshoot a single-page application using a combination of AWS serverless and storage services. The activity demonstrates how Amazon S3, API Gateway, Lambda, and CloudWatch work together in a modern web application architecture.

Week 11: Expanding Serverless Solution: Multi-Service Integrations

Goal

This week, you will implement serverless authentication and build an AI-powered chatbot that leverages Amazon Kendra and Amazon Bedrock to provide accurate responses from private organizational data.

Learning objectives

- Learn about advanced AWS services including Amazon Bedrock, Amazon Comprehend, Amazon Pinpoint, and Amazon Translate
- Implement serverless authentication
- Build a serverless chatbot in SimuLearn using private data

Outline

1. **Expanding Serverless Solution: Multi-Service Integrations**
    a. **Hands-On Practice**
        i. AWS SimuLearn: Serverless Authentication
        ii. AWS SimuLearn: Serverless Chatbot Using Private Data

Hands-on labs

- **AWS SimuLearn: Serverless Authentication (60 min)**
  In this hands-on lab, you will apply your knowledge of AWS Serverless Application Model (AWS SAM), Amazon Cognito, and AWS Amplify to deploy and connect backend and frontend services.
- **AWS SimuLearn: Serverless Chatbot Using Private Data (60 min)**
  In this hands-on lab, you will build a serverless chatbot application that answers questions using an organization's private data. You will also gain insight into how serverless event-driven architecture integrates AI services like Amazon Kendra and Amazon Bedrock to create powerful applications.