Task Statement 1: Implement Authentication and/or Authorization

Task 1: Configure Cognito for User Authentication

Goal: Set up an Amazon Cognito User Pool for authentication and a Hosted UI for sign-in/sign-up.

Steps:

- Create a User Pool with email as username.
- Set password policy and email verification.
- Enable a hosted UI (sign-up/sign-in).
- Test user registration and login.
- Retrieve and inspect the JWT token in browser/network tab.

Concepts covered: JWT, Cognito user pool, bearer token, least privilege.

Task 2: Add Identity Pool for Federated Access

Goal: Use a Cognito Identity Pool to assign IAM roles to authenticated and unauthenticated users.

Steps:

- Create an Identity Pool linked to the User Pool.
- Attach IAM roles to the Identity Pool.
- Use the temporary AWS credentials from STS to access S3.

Concepts covered: Identity federation, STS, role assumption, IAM roles.

Task 3: Implement IAM Role-Based Access (RBAC)

Goal: Secure an S3 bucket so only users assuming a specific role can access it.

Steps:

- Create a new IAM role with restricted S3 access.
- Assume the role using AWS CLI or SDK (sts:AssumeRole).
- Test the S3 access after assuming the role.

Concepts covered: Assume role, RBAC, IAM policies, principle of least privilege.

Task 4: Create and Compare IAM Policies

Goal: Understand the difference between AWS managed, customer-managed, and inline policies.

Steps:

- Attach an AWS managed policy (e.g., AmazonS3ReadOnlyAccess) to a user.
- Create a customer-managed policy that gives PutObject on a specific bucket.
- Create an inline policy for a Lambda function that allows DynamoDB:GetItem.

Concepts covered: IAM policy types, least privilege, resource policies.

Task 5: Use Resource-Based Policies

Goal: Grant cross-account or fine-grained access using resource-based policies.

Steps:

- Add a resource policy to an S3 bucket to allow public read or allow access to a specific role/user.
- Add a Lambda resource-based policy to allow invocation from an external account (mock).

Concepts covered: Resource-based vs identity-based policies, cross-service access.

Task 6: Make Authenticated AWS Calls Using Token

Goal: Use Cognito tokens to make secure API calls to API Gateway.

Steps:

- Create a simple REST API in API Gateway.
- Secure it with a Cognito Authorizer.
- Call the API using Postman with the retrieved JWT token.

Concepts covered: API authorization, JWT, bearer tokens, Cognito authorizer.

Bonus Task: Compare ACLs vs IAM vs Bucket Policies

Goal: Learn when to use each access mechanism.

Steps:

- Enable object ACLs on a bucket and make one object publicly readable.
- Attach a bucket policy that denies access from non-secure transport (non-HTTPS).
- Use IAM to grant user-specific bucket access.

Concepts covered: ACLs, IAM, S3 bucket policies, layered security.