<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>AWS Assignment Template</title>

<script src="https://cdnjs.cloudflare.com/ajax/libs/html2pdf/0.10.1/html2pdf.bundle.min.js"></script>

<style>

body {

font-family: 'Calibri', Arial, sans-serif;

max-width: 900px;

margin: 0 auto;

padding: 20px;

line-height: 1.6;

background: #f5f5f5;

}

.container {

background: white;

padding: 40px;

border-radius: 5px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

}

h1 {

text-align: center;

color: #232f3e;

border-bottom: 3px solid #FF9900;

padding-bottom: 10px;

}

h2 {

color: #232f3e;

background: #f0f0f0;

padding: 10px;

margin-top: 30px;

border-left: 4px solid #FF9900;

}

.question-block {

margin: 30px 0;

padding: 20px;

border: 1px solid #ddd;

border-radius: 5px;

background: #fafafa;

}

.question-title {

font-weight: bold;

color: #232f3e;

margin-bottom: 15px;

font-size: 16px;

}

.fill-blank {

background: #fffacd;

padding: 15px;

border-left: 4px solid #FF9900;

margin: 15px 0;

min-height: 100px;

border-radius: 3px;

}

.instruction {

color: #666;

font-style: italic;

font-size: 14px;

padding: 10px;

background: #e8f4f8;

border-radius: 3px;

margin: 10px 0;

}

.code-block {

background: #f4f4f4;

border: 1px solid #ddd;

padding: 12px;

border-radius: 3px;

font-family: 'Courier New', monospace;

font-size: 12px;

overflow-x: auto;

margin: 10px 0;

}

.button-group {

text-align: center;

margin: 30px 0;

gap: 10px;

}

button {

padding: 12px 30px;

margin: 5px;

font-size: 16px;

border: none;

border-radius: 5px;

cursor: pointer;

background: #FF9900;

color: white;

font-weight: bold;

}

button:hover {

background: #ec7211;

}

.note {

color: #d9534f;

font-weight: bold;

margin: 15px 0;

}

table {

width: 100%;

border-collapse: collapse;

margin: 15px 0;

}

table, th, td {

border: 1px solid #ddd;

}

th, td {

padding: 10px;

text-align: left;

}

th {

background: #232f3e;

color: white;

}

</style>

</head>

<body>

<div class="container" id="content">

<h1>AWS DATA ANALYTICS ASSIGNMENT</h1>

<p style="text-align: center; color: #666; margin-bottom: 30px;">

Complete Solutions Document | Semester Assignment

</p>

<!-- QUESTION 1 -->

<div class="question-block">

<div class="question-title">QUESTION 1: AWS Regions, Availability Zones, and Edge Locations</div>

<div class="instruction">📌 WHAT TO DO: Write 3-4 paragraph explanation. No screenshots needed. NO CODE.</div>

<p><strong>Question:</strong> Explain the difference between AWS Regions, Availability Zones, and Edge Locations. Why is this important for data analysis and latency-sensitive applications?</p>

<div class="fill-blank">

<strong>YOUR ANSWER:</strong><br><br>

<p><strong>AWS Regions:</strong></p>

[Write your explanation here - What is a region? Give example]

<br><br>

<p><strong>Availability Zones (AZs):</strong></p>

[Write your explanation here - What is AZ? Why multiple?]

<br><br>

<p><strong>Edge Locations:</strong></p>

[Write your explanation here - What are they? CDN usage?]

<br><br>

<p><strong>Why Important for Analytics:</strong></p>

[Write your explanation here - Low latency, compliance, disaster recovery]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED for this question</p>

</div>

<!-- QUESTION 2 -->

<div class="question-block">

<div class="question-title">QUESTION 2: List AWS Regions Using AWS CLI</div>

<div class="instruction">📌 WHAT TO DO: Run AWS CLI command, take screenshot of output, paste in Word</div>

<p><strong>Question:</strong> Using the AWS CLI, list all available AWS regions. Share the command used and the output.</p>

<div class="fill-blank">

<strong>COMMAND USED:</strong><br>

<div class="code-block">aws ec2 describe-regions --query 'Regions[\*].[RegionName,Endpoint]' --output table</div>

<strong>OUTPUT (Paste screenshot here):</strong><br>

[PASTE SCREENSHOT OF COMMAND OUTPUT]

<br><br>

<strong>EXPLANATION:</strong><br>

[Write - What does this command do? What information does it show?]

</div>

<p class="note">✅ SCREENSHOT: PASTE command output here</p>

</div>

<!-- QUESTION 3 -->

<div class="question-block">

<div class="question-title">QUESTION 3: Create IAM User with S3 Least Privilege</div>

<div class="instruction">📌 WHAT TO DO: Create IAM user in AWS Console, attach S3 read-only policy, take screenshots</div>

<p><strong>Question:</strong> Create a new IAM user with least privilege access to Amazon S3. Share the attached policies (JSON or screenshot).</p>

<div class="fill-blank">

<strong>USER CREATED:</strong><br>

Username: [Enter username you created]

<br><br>

<strong>PERMISSIONS ATTACHED:</strong><br>

Policy Name: [Which policy did you attach?]

<br><br>

<strong>POLICY SCREENSHOT:</strong><br>

[PASTE SCREENSHOT of policy details]

<br><br>

<strong>ACCESS KEYS SCREENSHOT:</strong><br>

[PASTE SCREENSHOT showing access keys created (blur sensitive parts)]

<br><br>

<strong>WHY LEAST PRIVILEGE?</strong><br>

[Write explanation - Why is this important? What can/cannot user do?]

</div>

<p class="note">✅ SCREENSHOTS: 2 needed (Policy + Access Keys)</p>

</div>

<!-- QUESTION 4 -->

<div class="question-block">

<div class="question-title">QUESTION 4: Compare S3 Storage Classes</div>

<div class="instruction">📌 WHAT TO DO: Write comparison. No screenshots. NO CODE.</div>

<p><strong>Question:</strong> Compare different Amazon S3 storage classes (Standard, Intelligent-Tiering, Glacier). When should each be used in data analytics workflows?</p>

<div class="fill-blank">

<strong>S3 STANDARD:</strong><br>

Cost: [High/Medium/Low] | Speed: [Fast/Medium/Slow]<br>

Use Case: [Your explanation]

<br><br>

<strong>INTELLIGENT-TIERING:</strong><br>

Cost: [High/Medium/Low] | Speed: [Fast/Medium/Slow]<br>

Use Case: [Your explanation]

<br><br>

<strong>GLACIER:</strong><br>

Cost: [High/Medium/Low] | Speed: [Fast/Medium/Slow]<br>

Use Case: [Your explanation]

<br><br>

<strong>DATA ANALYTICS WORKFLOW EXAMPLE:</strong><br>

[Write how you would use all 3 together in a real pipeline]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED</p>

</div>

<!-- QUESTION 5 -->

<div class="question-block">

<div class="question-title">QUESTION 5: Create S3 Bucket, Upload Dataset, Enable Versioning</div>

<div class="instruction">📌 WHAT TO DO: Create bucket, upload CSV file, enable versioning, upload same file again, take 2 screenshots</div>

<p><strong>Question:</strong> Create an S3 bucket and upload a sample dataset (CSV or JSON). Enable versioning and show at least two versions of one file.</p>

<div class="fill-blank">

<strong>BUCKET NAME:</strong><br>

[Enter your bucket name]

<br><br>

<strong>REGION:</strong><br>

[Select region - e.g., ap-south-1]

<br><br>

<strong>SAMPLE DATA (CSV) - What you uploaded:</strong><br>

<div class="code-block">

Date,Product,Sales,Region

2024-01-01,Laptop,50000,North

2024-01-02,Mobile,35000,South

2024-01-03,Laptop,48000,East

</div>

<strong>VERSIONING ENABLED SCREENSHOT:</strong><br>

[PASTE SCREENSHOT showing versioning turned ON]

<br><br>

<strong>FILE VERSIONS SCREENSHOT:</strong><br>

[PASTE SCREENSHOT showing 2+ versions of uploaded file]

<br><br>

<strong>EXPLANATION:</strong><br>

[Write - Why is versioning important? What can we do with it?]

</div>

<p class="note">✅ SCREENSHOTS: 2 needed (Versioning enabled + Multiple versions)</p>

</div>

<!-- QUESTION 6 -->

<div class="question-block">

<div class="question-title">QUESTION 6: Create Lifecycle Policy</div>

<div class="instruction">📌 WHAT TO DO: Create lifecycle rule in S3, move to Glacier after 30 days, delete after 90 days, screenshot the rule</div>

<p><strong>Question:</strong> Write and apply a lifecycle policy to move files to Glacier after 30 days and delete them after 90 days. Share the policy JSON or screenshot.</p>

<div class="fill-blank">

<strong>LIFECYCLE RULE NAME:</strong><br>

[Enter rule name you created]

<br><br>

<strong>LIFECYCLE RULE SCREENSHOT:</strong><br>

[PASTE SCREENSHOT of configured rule]

<br><br>

<strong>POLICY JSON:</strong><br>

<div class="code-block">

{

"Rules": [

{

"Id": "auto-archive-files",

"Status": "Enabled",

"Transitions": [

{

"Days": 30,

"StorageClass": "GLACIER"

}

],

"Expiration": {

"Days": 90

}

}

]

}

</div>

<strong>HOW IT WORKS:</strong><br>

[Write explanation - Days 1-30 what happens? Days 31-90? Day 91+?]

</div>

<p class="note">✅ SCREENSHOT: 1 needed (Lifecycle rule configuration)</p>

</div>

<!-- QUESTION 7 -->

<div class="question-block">

<div class="question-title">QUESTION 7: Compare RDS, DynamoDB, and Redshift</div>

<div class="instruction">📌 WHAT TO DO: Write comparison with one use case each. No screenshots. NO CODE.</div>

<p><strong>Question:</strong> Compare RDS, DynamoDB, and Redshift for use in different stages of a data pipeline. Give one use case for each.</p>

<div class="fill-blank">

<strong>RDS (Relational Database):</strong><br>

Type: [SQL/NoSQL/Other] | Purpose: [Your explanation]<br>

Use Case: [Real-world example]

<br><br>

<strong>DYNAMODB (NoSQL):</strong><br>

Type: [SQL/NoSQL/Other] | Purpose: [Your explanation]<br>

Use Case: [Real-world example]

<br><br>

<strong>REDSHIFT (Data Warehouse):</strong><br>

Type: [SQL/NoSQL/Other] | Purpose: [Your explanation]<br>

Use Case: [Real-world example]

<br><br>

<strong>DATA PIPELINE FLOW:</strong><br>

[Write - In what order would you use these three? Why?]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED</p>

</div>

<!-- QUESTION 8 -->

<div class="question-block">

<div class="question-title">QUESTION 8: Create DynamoDB Table + Lambda Function</div>

<div class="instruction">📌 WHAT TO DO: Create DynamoDB table, insert 3 records, create Lambda function, test it, take 3 screenshots</div>

<p><strong>Question:</strong> Create a DynamoDB table and insert 3 records manually. Then write a Lambda function that adds records when triggered by S3 uploads.</p>

<div class="fill-blank">

<strong>DYNAMODB TABLE NAME:</strong><br>

[Enter table name]

<br><br>

<strong>PRIMARY KEY:</strong><br>

[Enter partition key name and type]

<br><br>

<strong>SCREENSHOT 1: DynamoDB Table with 3 Records</strong><br>

[PASTE SCREENSHOT showing 3 manually inserted items]

<br><br>

<strong>LAMBDA FUNCTION NAME:</strong><br>

[Enter function name]

<br><br>

<strong>LAMBDA TRIGGER:</strong><br>

Service: S3 | Event: Object Created

<br><br>

<strong>LAMBDA CODE:</strong><br>

<div class="code-block">

import json

import boto3

import uuid

from datetime import datetime

dynamodb = boto3.resource('dynamodb')

table = dynamodb.Table('analytics-events')

def lambda\_handler(event, context):

try:

for record in event['Records']:

bucket = record['s3']['bucket']['name']

key = record['s3']['object']['key']

item = {

'event\_id': str(uuid.uuid4()),

'bucket': bucket,

'file': key,

'timestamp': datetime.now().isoformat()

}

table.put\_item(Item=item)

return {

'statusCode': 200,

'body': json.dumps('Records inserted successfully')

}

except Exception as e:

return {

'statusCode': 500,

'body': json.dumps(f'Error: {str(e)}')

}

</div>

<strong>SCREENSHOT 2: Lambda Code Deployed</strong><br>

[PASTE SCREENSHOT of Lambda function code]

<br><br>

<strong>SCREENSHOT 3: Lambda Execution Result</strong><br>

[PASTE SCREENSHOT showing successful execution]

<br><br>

<strong>SCREENSHOT 4: New Record in DynamoDB</strong><br>

[PASTE SCREENSHOT showing new item added by Lambda]

</div>

<p class="note">✅ SCREENSHOTS: 4 needed (DynamoDB records + Lambda code + Execution + New record)</p>

</div>

<!-- QUESTION 9 -->

<div class="question-block">

<div class="question-title">QUESTION 9: Serverless Computing & Lambda</div>

<div class="instruction">📌 WHAT TO DO: Write explanation of serverless. No screenshots. NO CODE.</div>

<p><strong>Question:</strong> What is serverless computing? Discuss pros and cons of using AWS Lambda for data pipelines.</p>

<div class="fill-blank">

<strong>WHAT IS SERVERLESS COMPUTING?</strong><br>

[Write definition - What does it mean? How is it different from traditional servers?]

<br><br>

<strong>PROS OF LAMBDA:</strong><br>

1. [First advantage with explanation]<br>

2. [Second advantage with explanation]<br>

3. [Third advantage with explanation]

<br><br>

<strong>CONS OF LAMBDA:</strong><br>

1. [First disadvantage with explanation]<br>

2. [Second disadvantage with explanation]<br>

3. [Third disadvantage with explanation]

<br><br>

<strong>BEST USE CASES FOR DATA PIPELINES:</strong><br>

[List what tasks are good for Lambda]

<br><br>

<strong>NOT SUITABLE FOR:</strong><br>

[List what tasks are NOT good for Lambda]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED</p>

</div>

<!-- QUESTION 10 -->

<div class="question-block">

<div class="question-title">QUESTION 10: Lambda with CloudWatch Logging</div>

<div class="instruction">📌 WHAT TO DO: Create Lambda that logs to CloudWatch, upload file to S3 to trigger it, take screenshot of logs</div>

<p><strong>Question:</strong> Create a Lambda function triggered by S3 uploads that logs file name, size, and timestamp to CloudWatch. Share code and a log screenshot.</p>

<div class="fill-blank">

<strong>LAMBDA FUNCTION NAME:</strong><br>

[Enter function name]

<br><br>

<strong>S3 TRIGGER:</strong><br>

Bucket: [Your bucket] | Event: Object Created

<br><br>

<strong>LAMBDA CODE:</strong><br>

<div class="code-block">

import json

import boto3

import logging

from datetime import datetime

logger = logging.getLogger()

logger.setLevel(logging.INFO)

def lambda\_handler(event, context):

try:

for record in event['Records']:

bucket = record['s3']['bucket']['name']

key = record['s3']['object']['key']

size = record['s3']['object'].get('size', 'Unknown')

message = f"File uploaded - Bucket: {bucket}, Key: {key}, Size: {size} bytes, Time: {datetime.now()}"

logger.info(message)

return {'statusCode': 200, 'body': json.dumps('Logged successfully')}

except Exception as e:

logger.error(f"Error: {str(e)}")

return {'statusCode': 500, 'body': json.dumps(f'Error: {str(e)}')}

</div>

<strong>CLOUDWATCH LOGS SCREENSHOT:</strong><br>

[PASTE SCREENSHOT showing logs with file name, size, timestamp]

<br><br>

<strong>EXAMPLE LOG OUTPUT:</strong><br>

[Write what the log message shows]

</div>

<p class="note">✅ SCREENSHOTS: 1 needed (CloudWatch logs)</p>

</div>

<!-- QUESTION 11 -->

<div class="question-block">

<div class="question-title">QUESTION 11: AWS Glue for ETL</div>

<div class="instruction">📌 WHAT TO DO: Create Glue Crawler, run it on S3 data, create Glue Job to convert CSV to Parquet, screenshot job code</div>

<p><strong>Question:</strong> Use AWS Glue to crawl your S3 dataset, create a Data Catalog table, and run a Glue job to convert CSV data to parquet. Share job code and output location.</p>

<div class="fill-blank">

<strong>GLUE CRAWLER NAME:</strong><br>

[Enter crawler name]

<br><br>

<strong>DATA SOURCE (S3):</strong><br>

[Enter S3 bucket and folder path]

<br><br>

<strong>DATABASE CREATED:</strong><br>

[Enter database name in Glue Catalog]

<br><br>

<strong>GLUE JOB NAME:</strong><br>

[Enter job name]

<br><br>

<strong>GLUE JOB CODE:</strong><br>

<div class="code-block">

import sys

from awsglue.transforms import \*

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB\_NAME'])

sc = SparkContext()

glueContext = GlueContext(sc)

job = Job(glueContext)

job.init(args['JOB\_NAME'], args)

dyf = glueContext.create\_dynamic\_frame.from\_catalog(

database="analytics\_db",

table\_name="csv\_data"

)

glueContext.write\_dynamic\_frame.from\_options(

frame=dyf,

connection\_type="s3",

connection\_options={"path": "s3://output-bucket/parquet/"},

format="parquet"

)

job.commit()

</div>

<strong>JOB SCREENSHOT:</strong><br>

[PASTE SCREENSHOT of Glue Job configuration]

<br><br>

<strong>OUTPUT LOCATION:</strong><br>

[Enter S3 path where Parquet files were created]

</div>

<p class="note">✅ SCREENSHOTS: 1 needed (Glue Job code)</p>

</div>

<!-- QUESTION 12 -->

<div class="question-block">

<div class="question-title">QUESTION 12: Compare Kinesis Services</div>

<div class="instruction">📌 WHAT TO DO: Write comparison. No screenshots. NO CODE.</div>

<p><strong>Question:</strong> Explain the difference between Kinesis Data Streams, Kinesis Firehose, and Kinesis Data Analytics. Provide a real-world example of how each would be used.</p>

<div class="fill-blank">

<strong>KINESIS DATA STREAMS:</strong><br>

Purpose: [Your explanation]<br>

Real-world Use Case: [Example]

<br><br>

<strong>KINESIS FIREHOSE:</strong><br>

Purpose: [Your explanation]<br>

Real-world Use Case: [Example]

<br><br>

<strong>KINESIS DATA ANALYTICS:</strong><br>

Purpose: [Your explanation]<br>

Real-world Use Case: [Example]

<br><br>

<strong>HOW ALL THREE WORK TOGETHER:</strong><br>

[Write a complete real-world pipeline example]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED</p>

</div>

<!-- QUESTION 13 -->

<div class="question-block">

<div class="question-title">QUESTION 13: Redshift Columnar Storage</div>

<div class="instruction">📌 WHAT TO DO: Write explanation. No screenshots. NO CODE.</div>

<p><strong>Question:</strong> What is columnar storage and how does it benefit Redshift performance for analytics workloads?</p>

<div class="fill-blank">

<strong>WHAT IS COLUMNAR STORAGE?</strong><br>

[Write explanation - How is it different from row-based storage?]

<br><br>

<strong>BENEFITS FOR ANALYTICS:</strong><br>

1. [First benefit with explanation]<br>

2. [Second benefit with explanation]<br>

3. [Third benefit with explanation]

<br><br>

<strong>EXAMPLE QUERY:</strong><br>

[Write example - Why is columnar storage faster for this?]

<br><br>

<strong>PERFORMANCE IMPROVEMENT:</strong><br>

[How much faster? Give percentage or comparison]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED</p>

</div>

<!-- QUESTION 14 -->

<div class="question-block">

<div class="question-title">QUESTION 14: Load CSV to Redshift</div>

<div class="instruction">📌 WHAT TO DO: Create Redshift cluster, load CSV using COPY command, run query, screenshot results</div>

<p><strong>Question:</strong> Load a CSV file from S3 into Redshift using the COPY command. Share table schema, command used, and sample query output.</p>

<div class="fill-blank">

<strong>REDSHIFT CLUSTER NAME:</strong><br>

[Enter cluster name]

<br><br>

<strong>TABLE SCHEMA:</strong><br>

<div class="code-block">

CREATE TABLE sales\_data (

date DATE,

product VARCHAR(50),

sales DECIMAL(10,2),

region VARCHAR(20)

);

</div>

<strong>COPY COMMAND USED:</strong><br>

<div class="code-block">

COPY sales\_data

FROM 's3://your-bucket/sales\_data.csv'

IAM\_ROLE 'arn:aws:iam::ACCOUNT-ID:role/redshift-role'

CSV

IGNOREHEADER 1;

</div>

<strong>QUERY RUN:</strong><br>

<div class="code-block">

SELECT \* FROM sales\_data LIMIT 10;

</div>

<strong>QUERY RESULTS SCREENSHOT:</strong><br>

[PASTE SCREENSHOT of query output showing loaded data]

</div>

<p class="note">✅ SCREENSHOTS: 1 needed (Query results)</p>

</div>

<!-- QUESTION 15 -->

<div class="question-block">

<div class="question-title">QUESTION 15: AWS Glue Data Catalog & Schema-on-Read</div>

<div class="instruction">📌 WHAT TO DO: Write explanation. No screenshots. NO CODE.</div>

<p><strong>Question:</strong> What is the role of the AWS Glue Data Catalog in Athena? How does schema-on-read work?</p>

<div class="fill-blank">

<strong>AWS GLUE DATA CATALOG:</strong><br>

Definition: [What is it?]<br>

Purpose: [Why is it important?]

<br><br>

<strong>SCHEMA-ON-READ:</strong><br>

Concept: [What does it mean?]<br>

How it Works: [Explain the process]

<br><br>

<strong>BENEFITS FOR ANALYTICS:</strong><br>

[Write 2-3 benefits with explanations]

<br><br>

<strong>REAL-WORLD USE CASE:</strong><br>

[Write how a company would use this]

</div>

<p class="note">✅ SCREENSHOT: NONE NEEDED</p>

</div>

<!-- QUESTION 16 -->

<div class="question-block">

<div class="question-title">QUESTION 16: Create Athena Table from S3 Data</div>

<div class="instruction">📌 WHAT TO DO: Create table in Athena, run query on S3 data, screenshot results</div>

<p><strong>Question:</strong> Create an Athena table from S3 data using Glue Catalog. Run a query and share the SQL + result screenshot.</p>

<div class="fill-blank">

<strong>S3 DATA LOCATION:</strong><br>

[Enter S3 path to your CSV file]

<br><br>

<strong>CREATE TABLE SQL:</strong><br>

<div class="code-block">

CREATE EXTERNAL TABLE IF NOT EXISTS sales\_analysis (

`date` DATE,

product STRING,

sales DECIMAL(10,2),

region STRING

)

STORED AS CSV

LOCATION 's3://your-bucket/athena-data/'

TBLPROPERTIES ('skip.header.line.count'='1');

</div>

<strong>QUERY RUN:</strong><br>

<div class="code-block">

SELECT \* FROM sales\_analysis LIMIT 10;

</div>

<strong>QUERY RESULTS SCREENSHOT:</strong><br>

[PASTE SCREENSHOT showing query results]

<br><br>

<strong>EXPLANATION:</strong><br>

[Write - How does Athena read data directly from S3? No loading needed?]

</div>

<p class="note">✅ SCREENSHOTS: 1 needed (Query results)</p>

</div>

<!-- QUESTION 17 -->

<div class="question-block">

<div class="question-title">QUESTION 17: QuickSight for Business Intelligence</div>

<div class="instruction">📌 WHAT TO DO: Create dataset, create visualization, create dashboard, screenshot dashboard</div>

<p><strong>Question:</strong> Describe how Amazon QuickSight supports business intelligence in a serverless data architecture. What are SPICE and embedded dashboards?</p>

<div class="fill-blank">

<strong>QUICKSIGHT OVERVIEW:</strong>