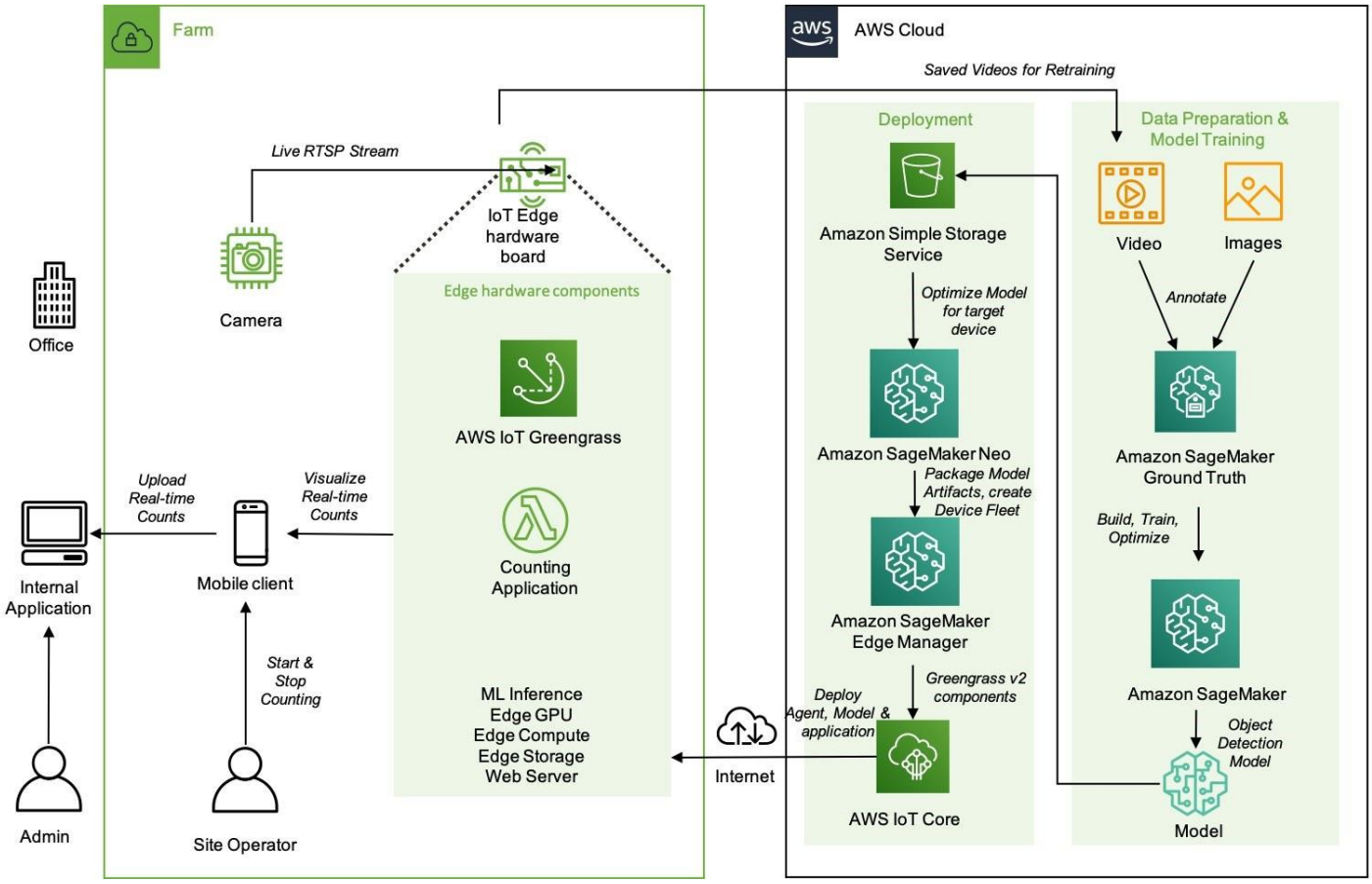
STUDENT REPORT

Aaliyah Raderberg / Coursera-Architect Solutions / AWS Cloud Solutions Architect Professional Certificate



Design Resilient Architecture

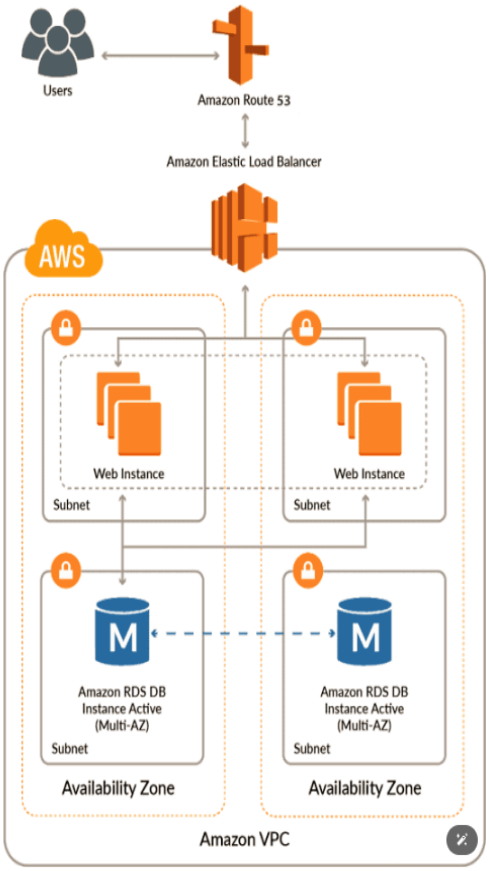


TABLE OF CONTENTS

INTRODUCTION

As part of this project, you will create a high-level architecture diagram that uses AWS service icons and arrows to depict an AWS solution for the given scenario. Create your diagram by using a tool like diagrams.net , or you can select a different tool by from the AWS Architecture Icons page by scrolling to the Drawing and diagramming tools section.

Scenario: You are working for a customer that runs their workloads on premises. Your customer has two workloads:

A three-tier architecture composed of a frontend (HTML, CSS, JavaScript), backend (Apache Web Server and a Java application), and database (MySQL). The three-tier application hosts a dynamic website that accepts user traffic from the internet.

A data analytics workload that runs Apache Hadoop. The analytics workload analyzes a massive amount of data that stored on premises and it also uses visualization tools to derive insights.

These components are currently running in the data center on physical servers. Currently, if a power outage occurred in the data center, all systems would be brought offline. Because of this issue (in addition to other benefits of the cloud), your customer wants to migrate all components to the cloud and, when possible, use AWS services to replace on-premises components.

Instructions

You have been tasked with designing a solution that uses AWS services to decouple the application layers (frontend, backend, and database), and that hosts both the application and the data analytics workload in the cloud. You can use managed services and advocate for refactoring the code to take advantage of cloud-native technologies, or you can do a lift and shift and advocate for minimal refactoring. Also, the data analytics solution currently runs on Hadoop and you have a requirement to spin up an Amazon EMR cluster for it. However, it's up to you to choose which AWS services you want to use for the ingestion, storage, and visualization of data.

Whichever architecture you choose to create, think about how the solution works and why you chose to use the services that you selected. Also, create an architecture diagram that depicts how both solutions will be hosted on AWS.

AWS Architecture: Web, App, and DB tier

METHOD

Three-Tier Architecture For Dynamic Website

To create a high-level architecture diagram for migrating the customer's workloads to AWS, we'll use the following AWS services:

Dynamic Website

Three-Tier Architecture for Dynamic Website

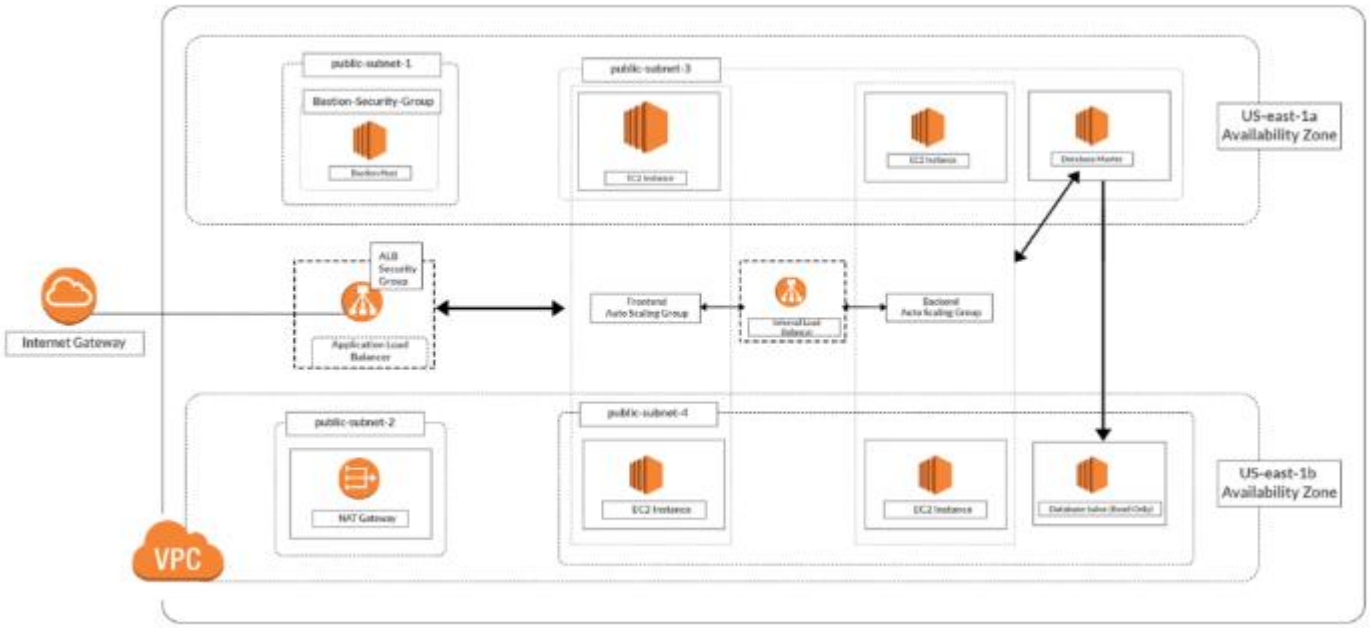
1. **Frontend:**
    - **Amazon S3**: To host the static website content (HTML, CSS, JavaScript).
    - **Amazon CloudFront**: To distribute content globally with low latency.
    - **Amazon Route 53**: For domain name resolution.
2. **Backend:**
    - **Amazon EC2**: Running Apache Web Server and the Java application. Alternatively, **AWS Elastic Beanstalk** can be used for easier deployment and scaling.
    - **Amazon API Gateway**: To expose RESTful APIs if necessary.
3. **Database:**
    - **Amazon RDS (MySQL)**: Managed relational database service.

Data Analytics Workload

1. **Data Storage:**
    - **Amazon S3**: For storing raw data and processed results.
2. **Data Processing:**
    - **Amazon EMR**: For running the Apache Hadoop cluster.
    - **AWS Glue**: For data cataloging and ETL jobs.
3. **Data Visualization:**
    - **Amazon QuickSight**: For visualizing the data and creating dashboards.
4. **Data Ingestion:**
    - **AWS DataSync**: For migrating data from on-premises to AWS.

- **AWS Snowball**: If large volumes of data need to be migrated.

Architecture Diagram



Below is the step-by-step design of the architecture diagram using diagrams.net:

- **Amazon S3** for static content storage.
- **Amazon CloudFront** in front of S3 for global distribution
- **Amazon Route 53** for DNS resolution
- **Amazon EC2** instances or **AWS Elastic Beanstalk** for the backend server.
- **Amazon RDS (MySQL)** for the database
- **Amazon EMR** for the Hadoop cluster
- **AWS Glue** for ETL jobs.
- **Amazon QuickSight** for data visualization.
- **AWS DataSync** for ongoing data migration.
- **AWS Snowball** for initial large data transfer.

EXPLANATION

Here's a visual representation of the architecture:

Explanation of Choices

- **Amazon S3 and CloudFront** are chosen for the frontend to ensure high availability, scalability, and low-latency access to static content.
- **Amazon EC2/Elastic Beanstalk** provides flexibility and scalability for the backend. Elastic Beanstalk simplifies the deployment process.
- **Amazon RDS** is selected for its managed service capabilities, automated backups, and ease of scaling.
- **Amazon EMR** is ideal for running Hadoop workloads, offering managed cluster services and integration with other AWS analytics services.
- **AWS Glue** and **Amazon QuickSight** streamline data processing and visualization, making it easier to gain insights from data.
- **AWS DataSync** and **AWS Snowball** provide efficient data migration from on-premises to the cloud, ensuring minimal downtime and data integrity during the transfer.

By using these AWS services, the solution ensures high availability, scalability, and reliability for both the web application and the data analytics workload.