**Part 2: Web Application Management Plan**

**University of Canada West**

Shilpi Kumari

**Table of Contents**

[**1.** **Introduction** 3](#_Toc193559637)

[**2.** **Cloud Cost Analysis:** 3](#_Toc193559638)

[**2.1 Cost Optimization Strategy** 4](#_Toc193559639)

[**3. Scalability and Monitoring Plan** 5](#_Toc193559640)

[**3.1 Autoscaling Strategy Using AWS Auto Scaling** 5](#_Toc193559641)

[**2.2 Monitoring Plan Using AWS CloudWatch** 9](#_Toc193559642)

[**3. High Availability Strategy** 10](#_Toc193559643)

[**3.1 High Availability Architecture Components** 10](#_Toc193559644)

[**3.2 Implementation Steps** 11](#_Toc193559645)

[**3.3 Benefits of the Strategy** 12](#_Toc193559646)

# **Introduction**

We make sure that there is always some way to reach a web application so as not to spoil the user experience. One high availability strategy implemented for maintaining the continuous operation of the Tour Planner Web Application was even in the failure of hardware, heightened traffic, and regional outages. Blue architecture by building on the AWS services like Autoscaling Groups, Multi-AZ deployment and Application Load Balancer, is architected to be resilient, fault tolerant and highly responsive under different type of load.

# **Cloud Cost Analysis:**

The installed infrastructure's expenses exist within the following table:

Table 1: Cloud costing Estimation Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **AWS Service Used** | **Estimated Monthly Cost** | **Details** |
| Web Server | EC2 (t2. micro) | ~$8 (if not on Free Tier) | Used to host the frontend and backend of the Tour Planner application. |
| Database Server | Amazon RDS (MySQL, db.t3. micro) | ~$15 | Managed MySQL instance for secure and scalable tour data storage. |
| Storage | Amazon S3 | ~$0.023/GB | Used for storing images and static assets. |
| Load Balancer | Application Load Balancer (ALB) | ~$18–$22 | Distributes traffic between multiple EC2 instances for high availability. |
| Monitoring | Amazon CloudWatch | ~$1–$3 (basic) | Logs, metrics, and alarms for EC2, RDS, and ALB monitoring. |
| Networking | VPC, Subnets, Security Groups | Free | Secure communication setup within AWS. |
| Security | IAM Roles & Security Groups | Free | Provides access control to services. |

Total Estimated Monthly Cost (without Free Tier): ~$45–$50

## **2.1 Cost Optimization Strategy**

Here are practical steps to **reduce AWS costs** while keeping performance intact:

**1. Use AWS Free Tier**

* Ensure both **EC2 t2. micro** and **RDS db.t3.micro** are within the **750 hours/month Free Tier**.
* Monitor usage regularly to avoid exceeding the free limits.

**2. Enable Auto Stop for EC2 & RDS in Off-Hours**

* If the Tour Planner app is not 24/7, schedule downtime (e.g., nights/weekends) using:
* Lambda + CloudWatch Events to **stop/start EC2 & RDS instances**.

**3. S3 Storage Optimization**

* Enable **S3 Lifecycle Rules** to transition older images to **S3 Infrequent Access (IA)** or **Glacier** for cheaper storage.

**4. CloudWatch Log Retention**

* Set log retention to **7 or 14 days** instead of the default **indefinite**.
* Delete old logs to avoid hidden CloudWatch costs.

**5. Consolidate EC2 Usage**

* Use **one EC2 instance** to run both frontend and backend if the traffic is low.
* Alternatively, use **Elastic Beanstalk** (with autoscaling) for better cost-performance balance.

**6. Review Load Balancer Usage**

* Use **target groups** wisely to avoid unnecessary instance registration.
* Use **ALB metrics** to auto-scale only when needed.

**7. Use Cost Explorer and Budgets**

* Set up **AWS Budgets** to get alerts when costs approach a certain threshold.
* Analyze **Cost Explorer** monthly to find idle or underused resources.

# **3. Scalability and Monitoring Plan**

## **3.1 Autoscaling Strategy Using AWS Auto Scaling**

To ensure **high availability** and **performance efficiency**, an Auto Scaling plan was defined and implemented for the **web server layer** (EC2 instances).

**Setup Overview:**

**Table 2: AWS Setup Overview**

|  |  |  |
| --- | --- | --- |
| **Component** | **AWS Service** | **Details** |
| Auto Scaling Group | EC2 Auto Scaling | Automatically increases/decreases EC2 instances based on CPU load or traffic |
| Load Balancer | Application Load Balancer (ALB) | Distributes incoming traffic across multiple instances for fault tolerance |
| Launch Template | EC2 Launch Template | Standardizes instance configuration and user data scripts |
| Scaling Policies | Target Tracking | Triggers scaling based on average **CPU Utilization ≥ 60%** |

**Scaling Parameters:**

* **Minimum Instances**: 1
* **Maximum Instances**: 3
* **Target CPU Utilization**: 60%
* **Cooldown Period**: 300 seconds (5 minutes)

This setup allows the infrastructure to grow during high traffic and scale down during idle times to optimize cost.

**Below are the screen shots:**

|  |
| --- |
| **EC2 – instance** |
| A screenshot of a computer  AI-generated content may be incorrect.  A screenshot of a computer  AI-generated content may be incorrect.  A screenshot of a computer  AI-generated content may be incorrect. |

|  |
| --- |
| **Load Balancer Configuration** |
|  |

|  |
| --- |
| **CloudWatch Monitoring** |
|  |

## **2.2 Monitoring Plan Using AWS CloudWatch**

To maintain application health and performance visibility, **CloudWatch** was configured for key components.

**Monitoring Setup:**

**Table 3: Monitoring Setup Parameters**

|  |  |  |
| --- | --- | --- |
| **Resource** | **Metrics Monitored** | **Alarm Configuration** |
| EC2 Instances | CPU Utilization, NetworkIn, NetworkOut, StatusCheckFailed | Alarms trigger if CPU > 70% for 5 mins |
| Load Balancer | Request Count, TargetResponseTime, HealthyHostCount | Alarm if TargetResponseTime > 2s |
| RDS (MySQL) | CPU Utilization, FreeStorageSpace, DatabaseConnections | Alarms on low disk space or high connection |
| CloudWatch Logs | EC2 syslogs, app logs (optional) | Optional alarms on error keyword patterns |

**Benefits of Monitoring Configuration:**

* **Real-Time Visibility**: Graphs show live traffic, usage spikes, or errors.
* **Early Detection**: CloudWatch Alarms alert admin of health check failures.
* **Scalability Intelligence**: Metrics inform scaling decisions dynamically.

# **3. High Availability Strategy**

**Objective**

The goal of the high availability strategy is to ensure that the Tour Planner web application remains **resilient**, **redundant**, and **accessible** even during failures, system upgrades, or unexpected traffic spikes.

## **3.1 High Availability Architecture Components**

Table 4: High Availability Architecture Components

|  |  |
| --- | --- |
| **Component** | **Configuration for High Availability** |
| EC2 Instances | Deployed in multiple Availability Zones (AZs) within the same region using an Auto Scaling Group. |
| Load Balancer | Application Load Balancer (ALB) distributes incoming traffic across multiple healthy EC2 instances. |
| Database (RDS) | Amazon RDS deployed in multi-AZ mode for automatic failover and redundancy. |
| S3 Bucket | Static files and images stored in Amazon S3 with cross-region durability and high availability. |
| VPC & Subnets | Two public and two private subnets across different AZs, ensuring network redundancy. |
| Health Checks | ALB continuously performs health checks to route traffic only to healthy backend instances. |

## **3.2 Implementation Steps**

1. **Auto Scaling Group with Multi-AZ EC2 Instances:**
   * Configured the Auto Scaling Group to launch EC2 instances in at least **two Availability Zones**.
   * Ensures that if one AZ goes down, another AZ keeps the application running.
2. **Application Load Balancer (ALB):**
   * Created and associated with the Auto Scaling Group.
   * Configured ALB with **listeners on port 80/443** and set to forward traffic to the target group of EC2 instances.
3. **Amazon RDS Multi-AZ Deployment:**
   * Enabled **multi-AZ** configuration during RDS creation.
   * Ensures failover support with an automatic standby in another AZ.
4. **S3 for Static Content:**
   * All static content (e.g., tour images) served from S3 to reduce load on web servers and ensure global availability.
5. **Security Groups & IAM Roles:**
   * Proper security groups are in place to ensure **secure communication** between EC2, RDS, and the ALB.
   * IAM roles control access to AWS resources securely.

## **3.3 Benefits of the Strategy**

**Table 5: Feature Advantages**

|  |  |
| --- | --- |
| **Benefit** | **Description** |
| Fault Tolerance | Ensures services remain operational during infrastructure or AZ failure. |
| Scalability | Auto Scaling allows the system to respond to increased or decreased traffic. |
| Improved Performance | ALB routes traffic to healthy and nearby resources for reduced latency. |
| Minimal Downtime | RDS multi-AZ ensures automatic failover without application-level changes. |
| Global Availability | S3 ensures static resources are highly available with low latency. |

* 1. **Architecture Design**

**Figure 1 – Architecture design of Tour Planner Web application**

**A screenshot of a computer

Description automatically generated**

**References**

* Amazon Web Services, Inc. (n.d.). AWS free tier. Amazon Web Services, Inc. [https://aws.amazon.com/free/?trk=31516735-77bb-4e6f-b031-2649837284cd&sc\_channel=ps&s\_kwcid=AL!4422!10!71880800488451!!!!71881348337014!!482210304!1150090143938060&ef\_id=234ff8407e301695bcf99f9f2d697b22:G:s&msclkid=234ff8407e301695bcf99f9f2d697b22&all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc&awsf.Free%20Tier%20Types=\*all&awsf.Free%20Tier%20Categories=\*all](https://aws.amazon.com/free/?trk=31516735-77bb-4e6f-b031-2649837284cd&sc_channel=ps&s_kwcid=AL!4422!10!71880800488451!!!!71881348337014!!482210304!1150090143938060&ef_id=234ff8407e301695bcf99f9f2d697b22:G:s&msclkid=234ff8407e301695bcf99f9f2d697b22&all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc&awsf.Free%20Tier%20Types=*all&awsf.Free%20Tier%20Categories=*all)
* **Amazon Web Services, Inc.** (n.d.). AWS Free Tier. Retrieved from <https://aws.amazon.com/free/>
* **Amazon Web Services, Inc.** (n.d.). AWS Auto Scaling. Retrieved from <https://aws.amazon.com/autoscaling/>
* **Amazon Web Services, Inc.** (n.d.). AWS CloudWatch Monitoring. Retrieved from <https://aws.amazon.com/cloudwatch/>
* **Amazon Web Services, Inc.** (n.d.). Amazon RDS Multi-AZ Deployments. Retrieved from <https://aws.amazon.com/rds/features/multi-az/>
* **Amazon Web Services, Inc.** (n.d.). AWS Application Load Balancer. Retrieved from <https://aws.amazon.com/elasticloadbalancing/application-load-balancer/>
* **Amazon Web Services, Inc.** (n.d.). AWS S3 Storage and Optimization. Retrieved from <https://aws.amazon.com/s3/>
* **AWS Well-Architected Framework** (2022). Best Practices for Cost Optimization in AWS Cloud. Retrieved from <https://docs.aws.amazon.com/wellarchitected/latest/framework/cost-optimization.html>
* **AWS Documentation** (n.d.). AWS Security Best Practices. Retrieved from [https://docs.aws.amazon.com/general/latest/gr/aws-security-best-practices.ht](https://docs.aws.amazon.com/general/latest/gr/aws-security-best-practices.html)

s