**Project Title: Deploying a Highly Available, Auto-Scaling WordPress Application for KicLabs Inc. on AWS**

**Problem Statement**

KicLabs Inc. a rapidly growing technology company, requires a robust WordPress website to manage its digital presence and customer interactions. The company faces the challenge of:

* **Handling fluctuating traffic volumes** without performance degradation.
* **Ensuring uninterrupted service availability** even if individual components fail.
* **Adapting to traffic spikes** with scalable infrastructure.
* **Maintaining media and content integrity** through reliable storage solutions.

**Project Scope**

The project involves the design and implementation of a scalable and highly available WordPress application infrastructure on AWS. This includes:

* Configuring AWS services such as VPC, EC2, RDS, and EFS.
* Ensuring integration and high availability.
* Documenting the setup and configuration process.

**Services**

1. **Amazon VPC (Virtual Private Cloud):** For network management and security.
2. **Amazon EC2 (Elastic Compute Cloud):** To host the WordPress application.
3. **Amazon RDS (Relational Database Service):** For managing the WordPress database.
4. **Amazon EFS (Elastic File System):** To store and share WordPress media files.

**Technical Goal**

The goal is to deliver a WordPress application that is:

* **Highly Available:** Ensuring minimal downtime through Multi-AZ deployments and auto-scaling.
* **Scalable:** Automatically adjusting to traffic loads.
* **Reliable:** Maintaining data integrity with persistent storage.
* **Secure:** Following best practices for cloud security.

**Architectural Diagram**

*Include a visual representation of your architecture here:*

**Components:**

* Amazon VPC with public and private subnets.
* EC2 Instances for WordPress in the public subnet.
* RDS Instance in the private subnet for MySQL.
* EFS File System for shared storage.
* Auto Scaling Group for EC2 instances.
* Load Balancer for distributing traffic.

**Technical Journal**

**Purpose:** Document the project development process, including design, implementation, and testing.

**Content:**

1. **Design Decisions:**

Explain architectural choices and configurations.

Justify the selection of services and instance types.

1. **Implementation Steps:**

Detail the setup process for each AWS service.

Provide step-by-step instructions or scripts used.

1. **Challenges and Solutions:**

Record issues encountered during the project and how they were resolved.

1. **Testing Results:**

Summarize results from performance and functionality tests.

*Documentation files can be included in a /docs directory.*

**Security Analysis**

**Purpose:** Ensure the infrastructure adheres to best security practices.

**Content:**

1. **Network Security:**

Analyze VPC and security group configurations.

1. **Data Security:**

Review encryption settings for RDS and EFS.

1. **Application Security:**

Ensure WordPress is securely configured.

1. **Compliance:**

Check adherence to AWS security best practices.

*Include any security analysis reports in a /security directory.*

**Submission**

**Deliverables:**

1. **Architectural Diagram:**

Visual representation of the deployed infrastructure.

File: diagram.png in /docs.

1. **Technical Journal:**

Detailed project documentation.

File: technical\_journal.md in /docs.

1. **Security Analysis Report:**

Report on security configurations and practices.

File: security\_analysis.md in /security.

1. **Configuration Files:**

Scripts and configuration files used in the setup.

Directory: /configs.

1. **Performance Testing Results:**

Evidence of site performance.

File: performance\_tests.md in /docs.

**Submission Guidelines:**

* **Format:** Markdown files (.md) for documentation and diagrams.
* **Repository Structure:**

/docs: Technical journal, performance testing results, and architectural diagrams.

/security: Security analysis report.

/configs: Configuration files.

* **Repository Name:** wordpress-aws-deployment-kiclabs
* **Submission Platform:** Push the project to GitHub and provide the repository link.

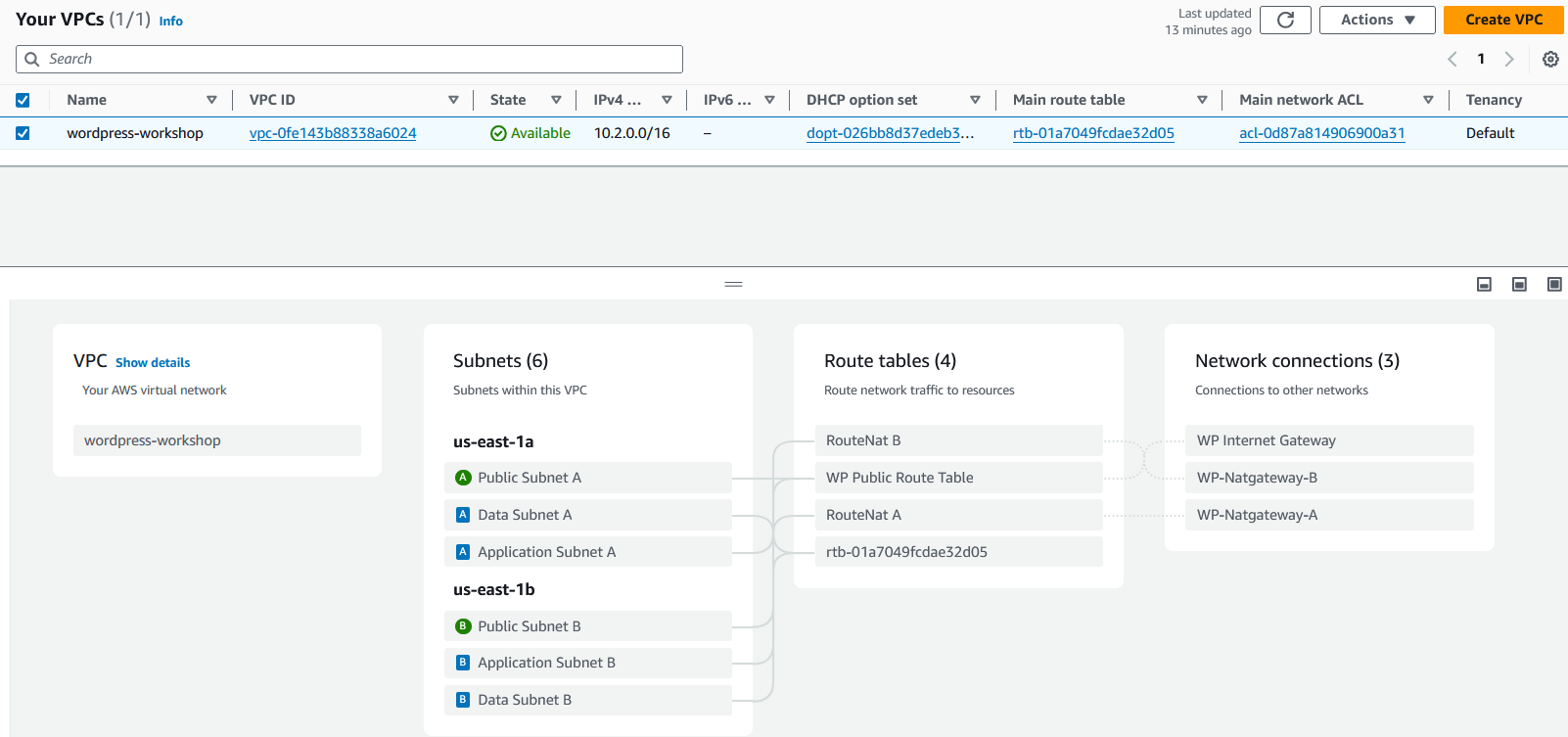
Hands – on Guide

<https://catalog.us-east-1.prod.workshops.aws/workshops/3de93ad5-ebbe-4258-b977-b45cdfe661f1/en-US/introduction>

**Laying the foundations**

**Lab 1: Configure the network.**

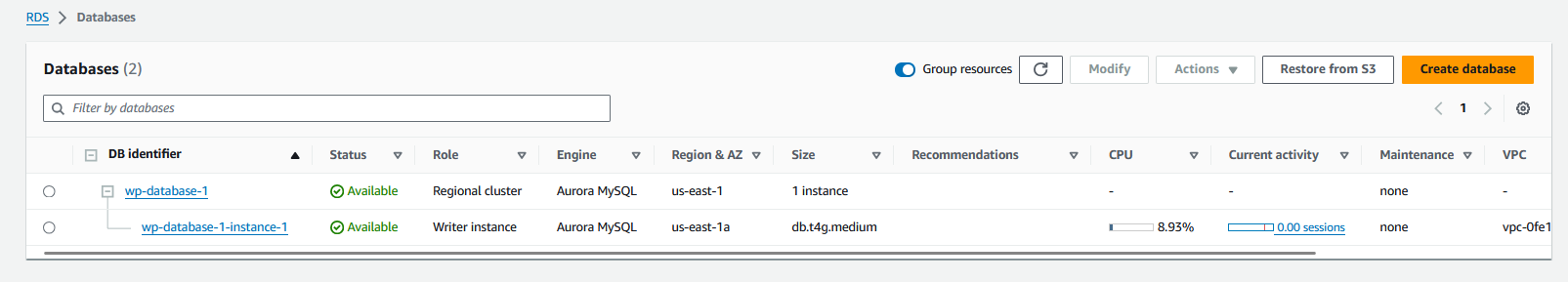
Resource map:



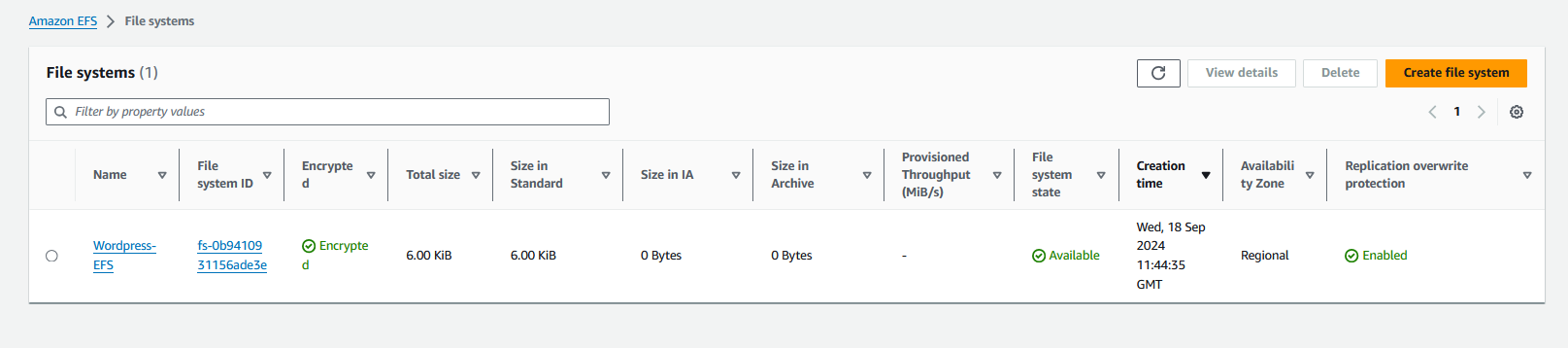
**Building the Data Tier**

**Lab 2: Set up the RDS database**

**Create database security groups:**

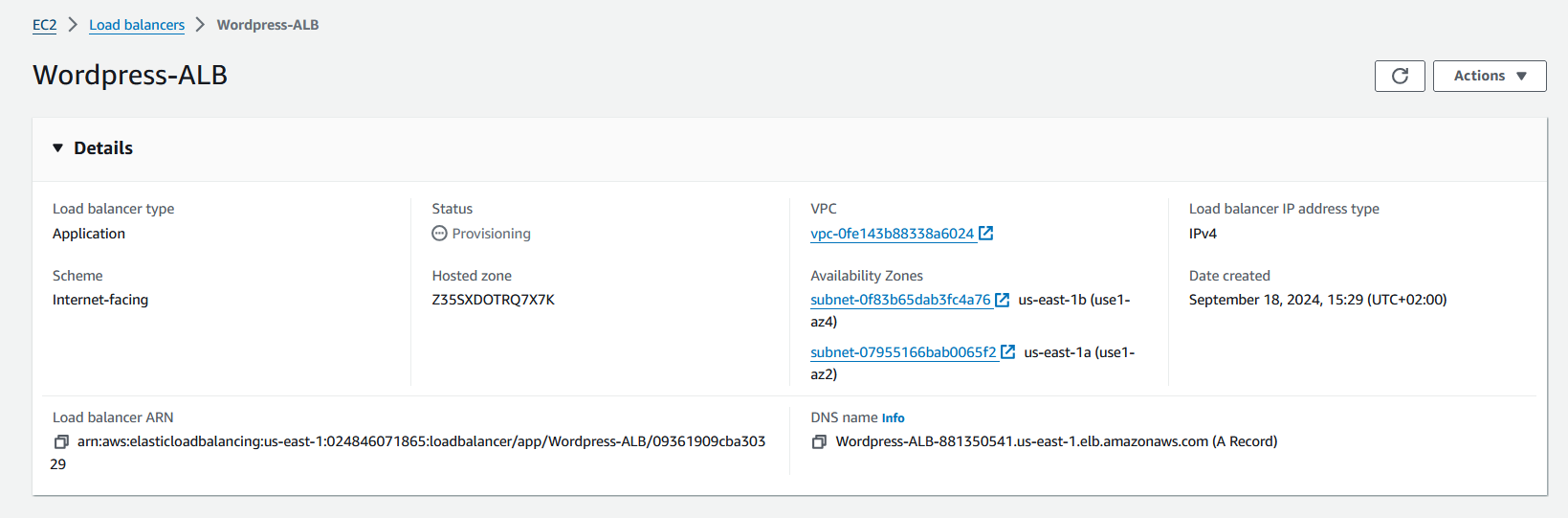
[](https://catalog.us-east-1.prod.workshops.aws/workshops/3de93ad5-ebbe-4258-b977-b45cdfe661f1/en-US/datatier/lab2" \l "create-database-security-groups)

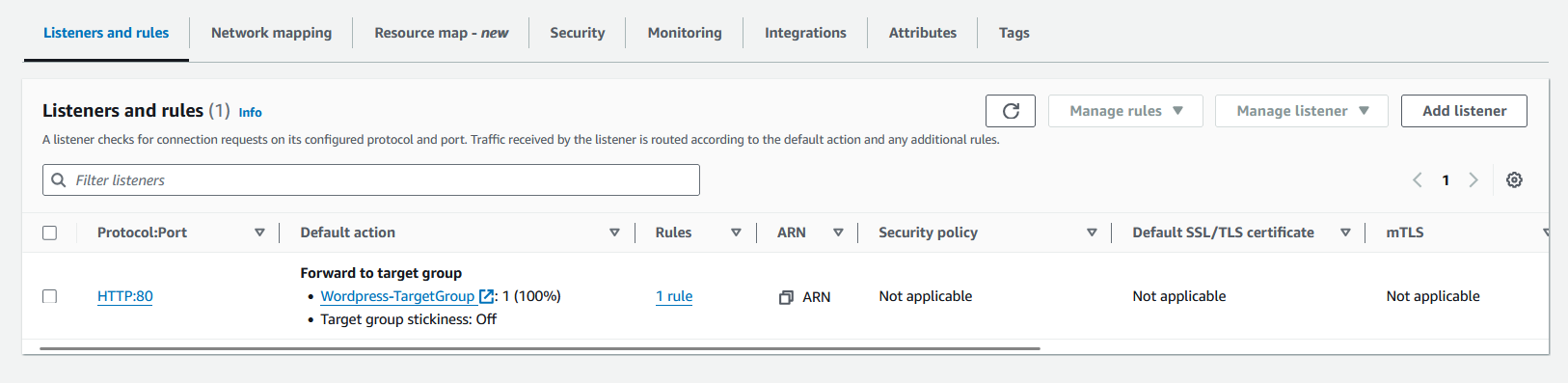
# **[Lab 3: Create the shared file system](https://catalog.us-east-1.prod.workshops.aws/workshops/3de93ad5-ebbe-4258-b977-b45cdfe661f1/en-US/datatier/lab2" \l "create-database-security-groups)**

[](https://catalog.us-east-1.prod.workshops.aws/workshops/3de93ad5-ebbe-4258-b977-b45cdfe661f1/en-US/datatier/lab2" \l "create-database-security-groups)

**Build the Application Tier:**

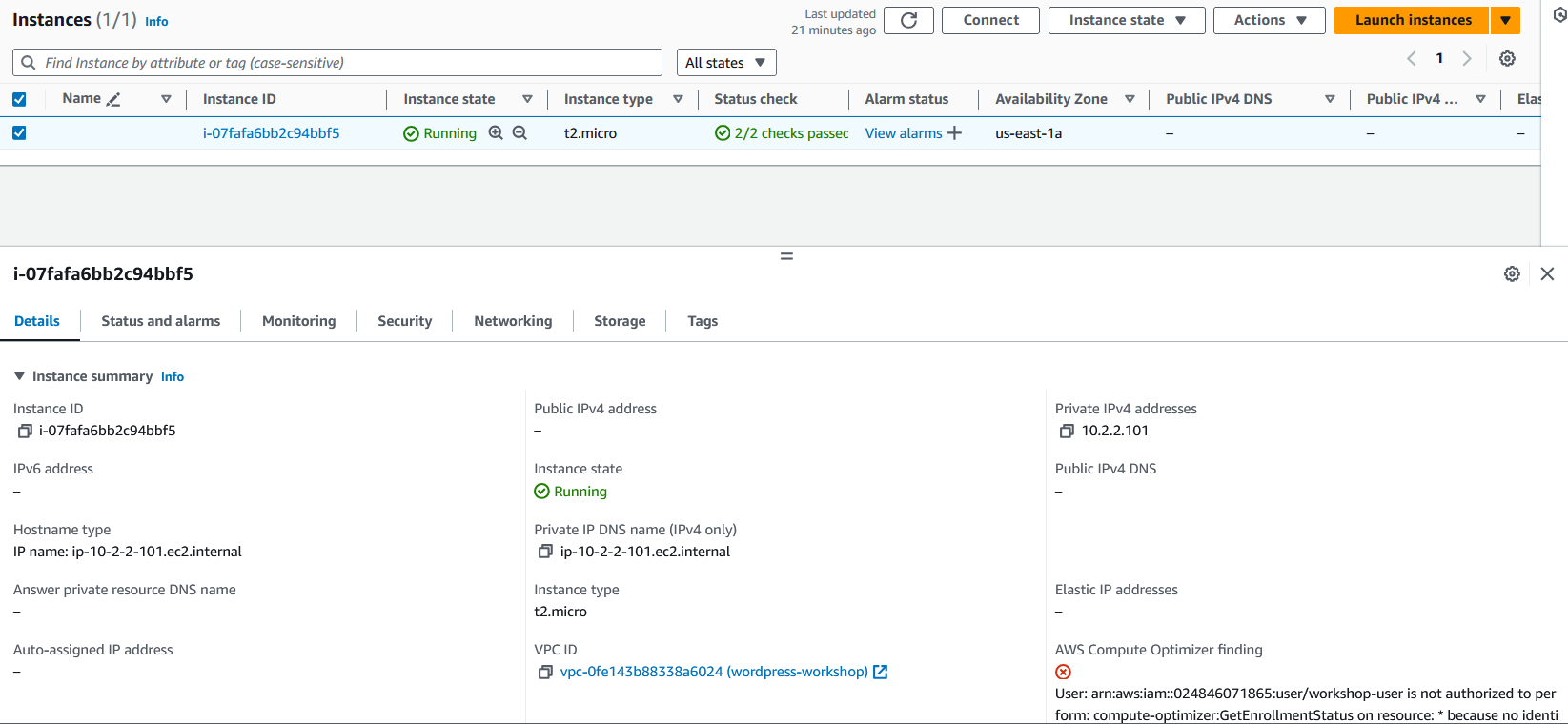
**Lab 4: Create the load balancer**





# **Lab 5: Create a launch Template**

**Launch an Instances:**

:

**Launch Template:**

