3.2. Student Handout

AWS Elastic Beanstalk (EB) and EB CLI Student Handout

Overview of AWS Elastic Beanstalk

AWS Elastic Beanstalk is a platform-as-a-service (PaaS) that simplifies the deployment and management of applications in the AWS Cloud. It abstracts the complexities of infrastructure management, allowing you to focus on your application.

Key Features:

- Supports multiple programming languages and frameworks.
- Automatically handles the deployment, from capacity provisioning, load balancing, and auto-scaling to application health monitoring.

Examples:

1. Deploy a Java web application using Elastic Beanstalk.
2. Use Elastic Beanstalk to manage a Node.js application with automatic scaling.
3. Deploy a Python Django application with integrated monitoring.

What is EB CLI?

The **EB CLI** (Elastic Beanstalk Command Line Interface) is a tool that allows you to interact with Elastic Beanstalk from your terminal, streamlining the deployment and management process.

Key Benefits:

- **Automation**: Automate deployment and management tasks.
- **Efficiency**: Perform operations directly from the command line.
- **Version Control**: Manage application versions and rollbacks.

Examples:

1. Deploy an application using `eb deploy` from the command line.
2. Retrieve logs using `eb logs` for debugging.

3. Scale an application using `eb scale <number_of_instances>`.

Installing and Configuring EB CLI

Step 1: Installing EB CLI

1. **Download AWS CLI**: Visit the AWS CLI documentation and download the installer for your OS.
2. **Install AWS CLI**: Follow the installation instructions.
3. **Verify Installation**: Run `aws --version` in your terminal.

Install EB CLI:

```
pip install awsebcli
```

Step 2: Configuring EB CLI

1. **Configure AWS CLI**: Run `aws configure` and enter your AWS credentials.
2. **Verify EB CLI Installation**: Run `eb --version`.

Examples:

1. Install AWS CLI on macOS and verify with `aws --version`.
2. Configure AWS CLI with your credentials using `aws configure`.
3. Install EB CLI using `pip install awsebcli`.

Configuring Elastic Beanstalk Environments and Applications via CLI

Creating an Application

1. Navigate to your project directory.
2. Run `eb init` to configure your application.

Creating an Environment

1. Run `eb create` to create a new environment.

Examples:

1. Initialize a Node.js application with `eb init`.
2. Create a production environment using `eb create`.
3. Set up a Python application with `eb init` and deploy it.

Managing Applications with EB CLI

Common Commands:

- **Deploy**: `eb deploy` to deploy your application.
- **Status**: `eb status` to view environment status.
- **Scale**: `eb scale <number_of_instances>` to adjust instance count.
- **Terminate**: `eb terminate` to delete an environment.

Examples:

1. Deploy a new version of your application using `eb deploy`.
2. Check the status of your environment with `eb status`.
3. Scale your application to 5 instances using `eb scale 5`.

Monitoring and Scaling Applications Using EB CLI Commands

Monitoring Application Health

- Run `eb health` to check the health status of your environment.

Scaling Your Application

- Use `eb scale <number_of_instances>` to manually scale your application.

Examples:

1. Monitor application health with `eb health`.
2. Scale down to 2 instances using `eb scale 2`.
3. Scale up to 10 instances for high traffic using `eb scale 10`.

Handling Environment Configurations, Logs, and Versions with EB CLI

Retrieving Logs

- Run `eb logs` to retrieve logs from your instances.

Managing Application Versions

- List versions: `eb appversion`.
- Deploy a specific version: `eb deploy --version <version_label>`.

Managing Environment Variables

- Set variables: `eb setenv VAR_NAME=value`.
- Retrieve variables: `eb printenv`.

Examples:

1. Retrieve logs for debugging with `eb logs`.
2. Deploy a previous version using `eb deploy --version v1.0.0`.
3. Set environment variables with `eb setenv API_KEY=12345`.

Hands-On: Deploying and Managing an Application on Elastic Beanstalk Using EB CLI

Steps:

1. **Initialize**: Run `eb init` in your project directory.
2. **Create Environment**: Run `eb create`.
3. **Deploy**: Run `eb deploy`.
4. **Monitor**: Run `eb health`.
5. **Retrieve Logs**: Run `eb logs`.
6. **Scale**: Run `eb scale 3`.
7. **Terminate**: Run `eb terminate`.

Examples:

1. Deploy a Django application using the steps above.
2. Manage a Ruby on Rails application with EB CLI.
3. Scale a PHP application to handle increased traffic.

Conclusion

In this session, you learned about:

- **AWS Elastic Beanstalk**: A managed service for deploying applications.
- **EB CLI**: A tool for managing Elastic Beanstalk applications from the command line.
- **Installation and Configuration**: How to set up the EB CLI.
- **Application Management**: Deploying, scaling, and monitoring applications.
- **Environment Management**: Handling logs, versions, and environment variables.

By mastering these tools and commands, you can efficiently deploy and manage applications on AWS Elastic Beanstalk.