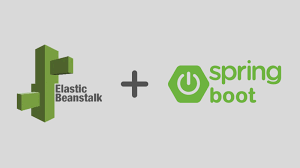
1)





Overview

A company wants to build a web application that provides services to its customers. The company chooses to use Spring Boot as the framework for its web application, and MySQL as the database to store its data. The company also decides to host its web application on AWS, and use Amazon RDS as its managed database service and Amazon EBS as its storage service.

Architecture

The architecture of the system consists of the following components:

Amazon EC2 instances: The company creates EC2 instances to host its Spring Boot application. These instances are configured with a security group that allows inbound traffic on port 80 and 443.

Amazon RDS: The company uses Amazon RDS to host its MySQL database. They create a database instance and configure it with the required security group and access control policies.

Amazon EBS: The company uses Amazon EBS to store its application data. They create an EBS volume and attach it to their EC2 instances.

Spring Boot application: The company develops its web application using Spring Boot, and configures it to connect to the MySQL database hosted on Amazon RDS. They also configure the application to store its data on the EBS volume attached to their EC2 instances.

Implementation

To implement the system, the company performs the following steps:

Create Amazon RDS instance: The company creates an Amazon RDS instance and configures it with the necessary parameters such as the database engine, instance type, and storage size. They also create a new database and a user with the necessary permissions to access the database.

Create Amazon EBS volume: The company creates an Amazon EBS volume and attaches it to their EC2 instances. They also format the volume and mount it on their instances.

Install and configure Spring Boot application: The company installs Spring Boot on their EC2 instances and configures it to connect to the MySQL database hosted on Amazon RDS. They also configure the application to store its data on the EBS volume attached to their instances.

Deploy Spring Boot application: The company deploys its Spring Boot application on their EC2 instances, and starts the application.

Benefits

Using Spring Boot with MySQL RDS and EBS in AWS provides the following benefits:

Scalability: The company can easily scale its web application by adding more EC2 instances, and increasing the storage size of its Amazon EBS volume.

Reliability: The use of Amazon RDS as the managed database service provides high availability and reliability for the company's database.

Cost-effective: Using AWS services such as Amazon RDS and Amazon EBS helps the company save costs, as they don't have to worry about managing their own infrastructure.

Easy management: AWS services such as Amazon RDS and Amazon EBS are easy to manage, as they provide a user-friendly interface for managing the database and storage resources.

Conclusion

Using Spring Boot with MySQL RDS and EBS in AWS provides a scalable, reliable, cost-effective, and easy-to-manage solution for building web applications. With the help of AWS services, companies can focus on building their applications without worrying about managing infrastructure.

Reference

https://aws.amazon.com/blogs/devops/deploying-a-spring-boot-application-on-aws-using-aws-elastic-beanstalk/

2)



Overview

A company wants to build a web application that provides services to its customers. The company chooses to use Spring Boot as the framework for its web application, and DynamoDB as the NoSQL database to store its data.

Architecture

The architecture of the system consists of the following components:

Amazon EC2 instances: The company creates EC2 instances to host its Spring Boot application. These instances are configured with a security group that allows inbound traffic on port 80 and 443.

DynamoDB: The company uses DynamoDB to host its NoSQL database. They create tables and configure them with the required read and write capacity.

Spring Boot application: The company develops its web application using Spring Boot, and configures it to connect to the DynamoDB tables.

Implementation

To implement the system, the company performs the following steps:

Create DynamoDB tables: The company creates DynamoDB tables and configures them with the necessary parameters such as the table name, primary key, and read and write capacity.

Install and configure Spring Boot application: The company installs Spring Boot on their EC2 instances and configures it to connect to the DynamoDB tables. They also configure the application to use the AWS SDK for Java to interact with DynamoDB.

Deploy Spring Boot application: The company deploys its Spring Boot application on their EC2 instances, and starts the application.

Benefits

Using Spring Boot with DynamoDB provides the following benefits:

Scalability: DynamoDB provides automatic scaling of read and write capacity, which makes it easy to scale the database as the application grows.

High performance: DynamoDB is a highly performant NoSQL database, which makes it suitable for applications that require fast read and write operations.

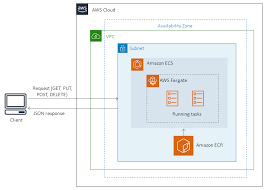
Low cost: DynamoDB is a serverless database, which means that the company only pays for the resources they use, which makes it a cost-effective solution for building web applications.

Easy management: DynamoDB is easy to manage, as it provides a user-friendly interface for managing the database and tables.

Conclusion

Using Spring Boot with DynamoDB provides a scalable, highly performant, cost-effective, and easy-to-manage solution for building web applications. With the help of DynamoDB, companies can focus on building their applications without worrying about managing infrastructure.

3) case study on using Spring Boot with ECS, Docker Hub, and RDS to build a doctor application:





Overview

A healthcare company wants to build a web application that helps patients find doctors in their area. The company chooses to use Spring Boot as the framework for its web application, and ECS as the container orchestration service to host its application. They also decide to use Docker Hub as the container registry service to store their application images and Amazon RDS as the managed database service to store their data.

Architecture

The architecture of the system consists of the following components:

Amazon EC2 instances: The company creates EC2 instances to host their ECS cluster. These instances are configured with a security group that allows inbound traffic on port 80 and 443.

Amazon RDS: The company uses Amazon RDS to host their MySQL database. They create a database instance and configure it with the required security group and access control policies.

Docker Hub: The company uses Docker Hub to store their Spring Boot application images.

ECS: The company uses ECS to orchestrate and manage their Docker containers. They create a task definition that specifies their Docker image, container port mappings, and resource requirements.

Spring Boot application: The company develops their web application using Spring Boot, and configures it to connect to the MySQL database hosted on Amazon RDS.

Implementation

To implement the system, the company performs the following steps:

Create Amazon RDS instance: The company creates an Amazon RDS instance and configures it with the necessary parameters such as the database engine, instance type, and storage size. They also create a new database and a user with the necessary permissions to access the database.

Develop and build Spring Boot application: The company develops their Spring Boot application and builds their Docker image. They also push their image to Docker Hub.

Create ECS cluster: The company creates an ECS cluster and configures it with the necessary parameters such as the number of instances, instance type, and security group.

Create ECS task definition: The company creates a task definition that specifies their Docker image, container port mappings, and resource requirements.

Deploy Spring Boot application: The company deploys their Spring Boot application on their ECS cluster by creating a service that references their task definition.

Benefits

Using Spring Boot with ECS, Docker Hub, and RDS provides the following benefits:

Scalability: The company can easily scale their web application by adding more instances to their ECS cluster and increasing the resources allocated to their task definition.

High availability: The use of ECS and RDS provides high availability for the company's web application and database.

Cost-effective: Using AWS services such as ECS and RDS helps the company save costs, as they don't have to worry about managing their own infrastructure.

Easy management: AWS services such as ECS and RDS are easy to manage, as they provide a user-friendly interface for managing the infrastructure.

Conclusion

Using Spring Boot with ECS, Docker Hub, and RDS provides a scalable, highly available, cost-effective, and easy-to-manage solution for building web applications. With the help of AWS services, companies can focus on building their applications without worrying about managing infrastructure. In the case of the healthcare company, they were able to build a doctor application that helped patients find doctors in their area, and deploy it on AWS with ease.