[AWS Cloud Architecture For Java Spring Boot Developers](https://tcsglobal.udemy.com/course/aws-java/)

Design, Deploy, Secure Applications with VPC. Subnets, Auto Scaling, Load Balancing, ECS Fargate, CI/CD Pipelines

**Description**

**Take your Java Spring Boot development skills to the cloud and build robust, scalable applications!**

This comprehensive course equips Java Spring Boot developers with the essential knowledge to design and deploy highly available, scalable, secure, and reliable applications on the AWS cloud platform. **No prior AWS experience is necessary!**

We'll begin with cloud computing fundamentals and progressively guide you through the core AWS services crucial for your success:

* **Compute:** Explore both EC2 instances and ECS Fargate for containerized deployments
* **Storage:** Explore object storage with S3 for efficient data management.
* **Identity & Access Management (IAM):** Implement robust security measures to protect your AWS resources.
* **Databases:** Leverage RDS for managing relational databases effectively.
* **Load Balancing:** Uncover the power of Application Load Balancers (ALB) to ensure high availability for your applications.
* **Content Delivery Network (CDN):** Optimize global content delivery with CloudFront.
* **DNS Services:** Learn to manage domain names effectively using Route 53.
* **Networking:** Gain a comprehensive understanding of VPCs, Subnets, NAT Gateways, and Route Tables to build secure and scalable network architectures.
* **Auto Scaling:** Implement automated scaling mechanisms to dynamically adjust resources based on application demands.

**Embrace modern CI/CD practices!** We'll delve into ***CodeBuild***, ***CodeCommit***, and ***CodePipeline*** to establish automated deployment workflows and ensure smooth rolling updates with zero downtime deployments.

**By the end of this course, you'll be equipped to:**

* Design and deploy highly available, scalable, and secure Cloud Infrastructure on AWS
* Deploy Java / Spring Boot application
* Implement containerization with ECS Fargate for efficient resource utilization
* Leverage CI/CD pipelines for automated application deployments and updates
* AutoScaling
* Manage infrastructure with core AWS services like S3, IAM, RDS, and more.
* Build a strong foundation for your future certification.

## Instructor

[**Vinoth Selvaraj**](https://tcsglobal.udemy.com/user/vinoth-selvaraj/)

Principal Engineer

Vinoth is a **Principal Engineer** with certifications as an **AWS Certified Solutions Architect Associate, Certified Kubernetes Application Developer**, and **Google Cloud Engineer**. He is passionate about software development, with a focus on Microservice Architecture, cloud computing, and developing scalable, high-performance distributed systems.

An avid enthusiast of **Java, Reactive Programming, Spring, Docker, Kubernetes, and Cloud Computing**, Vinoth expertly leverages these technologies to deliver robust and scalable solutions.

A screenshot of a computer

Description automatically generated

A group of containers with lids

Description automatically generated

A screenshot of a computer

Description automatically generated

Create Security group

**\*\*\* Resource \*\*\* - Docker Command For Postgres**

<https://github.com/vinsguru/aws-cloud-architect-essentials/blob/main/ec2/running-postgres.md>

So now we are going to launch 3 EC2 Instances

A screenshot of a computer

Description automatically generated

We have 3 instances are up and running

Logging in to db

A screenshot of a computer

Description automatically generated

Docker command to run postgress

A screenshot of a computer program

Description automatically generated

Now DB is ready to accept connections

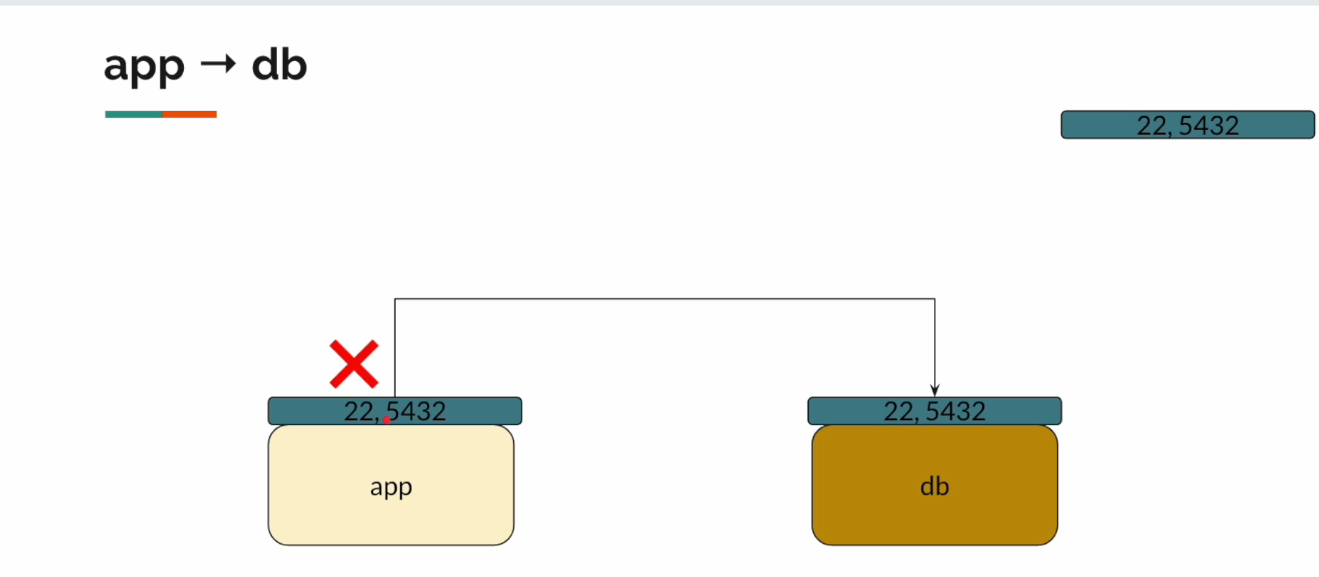
Not trying connecting to app instance

Not app is trying to connect to db

A screen shot of a computer

Description automatically generated

Sometimes people do like below, but it is of high risk



They open 5432 SG for all, but this should be avoided.

A screenshot of a computer

Description automatically generatedA diagram of a computer network

Description automatically generated with medium confidence

Terminated all instances and Removed all Security Groups.

13. Self referencing security groups.

Default SG has 1 default Inbound rule with all traffics allowed.

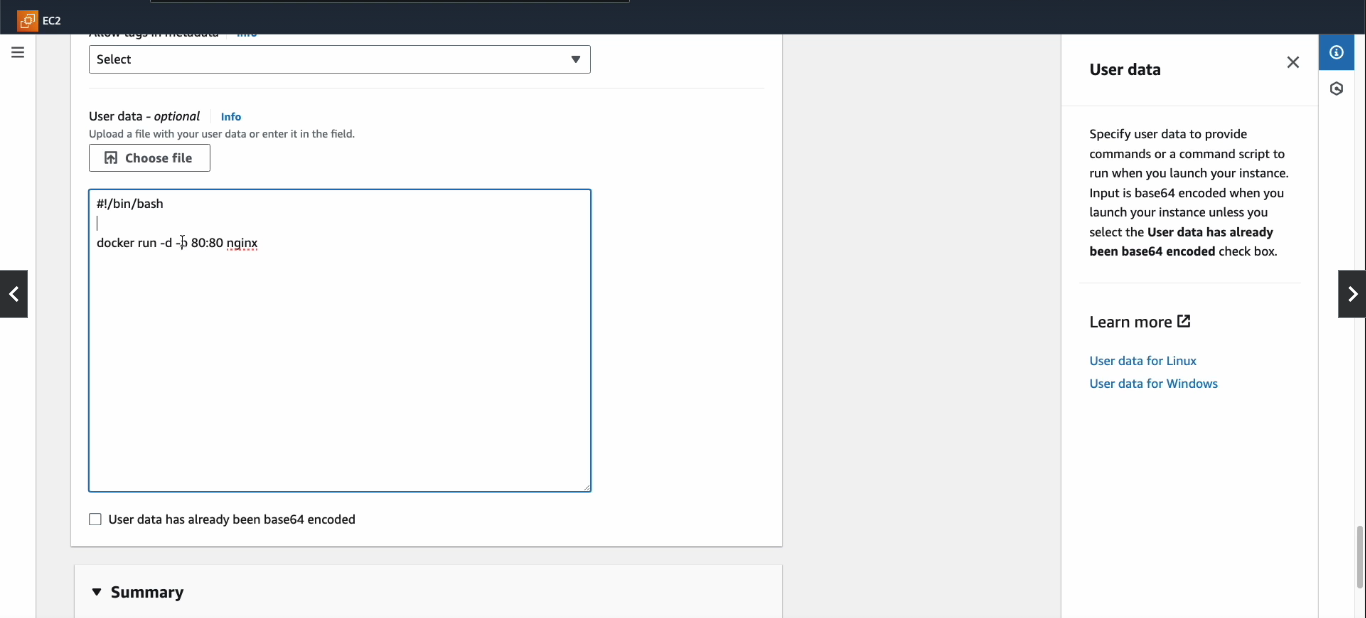
A screenshot of a computer

Description automatically generatedA diagram of a payment system

Description automatically generated

EC2 instance – User Data

Some commands or Scripts that we want to run once the system loads and gets ready.



A screenshot of a computer

Description automatically generated

Summary

A white background with black text

Description automatically generated

Region – Geographical location where we will be deploying our application.

A white background with black text

Description automatically generated A group of people with text

Description automatically generated with medium confidence

In Microservices architecture we can use self referencing SGs.