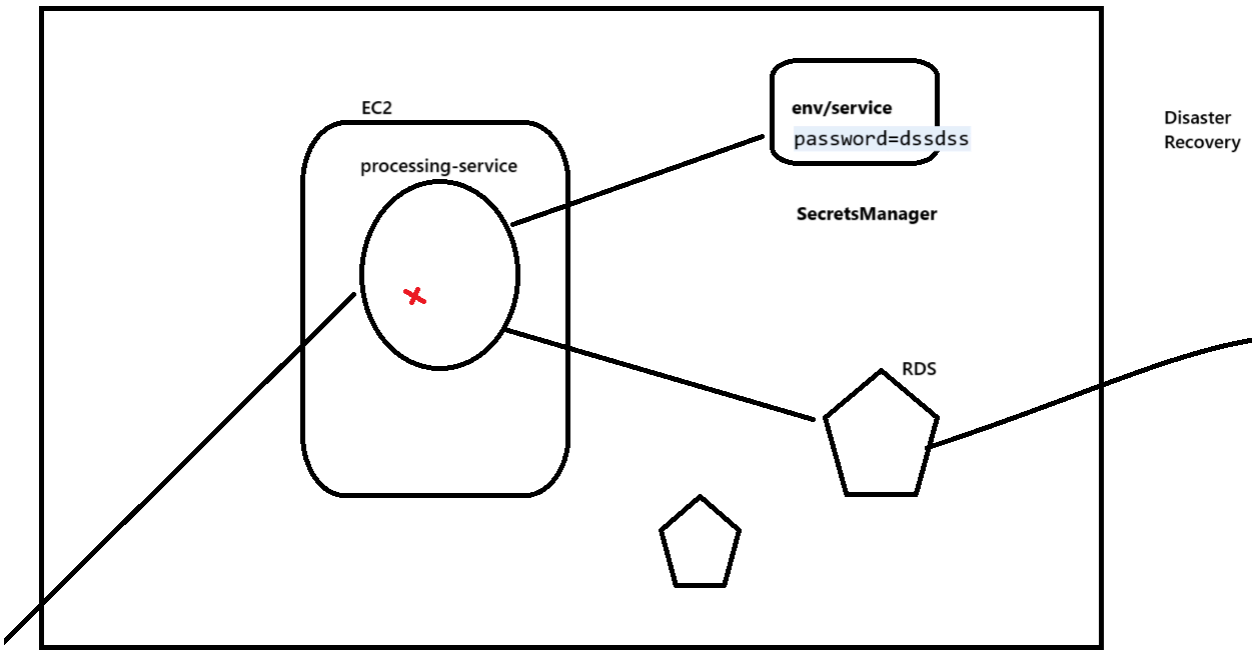
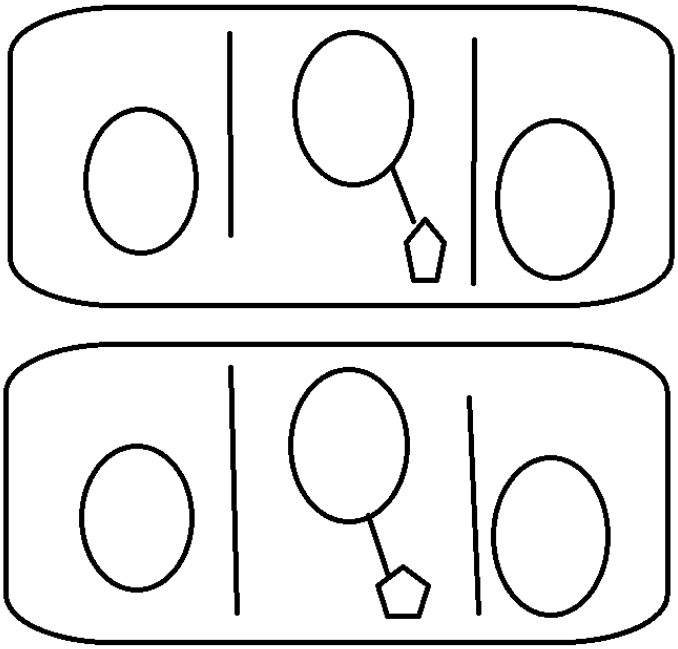
DIAGRAMS





Env Specific - service Specific
sensitive data

LIVE NOTES

3: Working with AWS

1. Setting up AWS instance EC2 (Spring Boot runs)
2. RDS (MySQL DB)
3. Secrets Manager - (storing sensitive data)

https://aws.amazon.com/console/

Client confirms to pick region where AWS Datacenter should be allocated.

Amazon Linux 2

Core & RAM
2Core - 4GB RAM

key pair:
    securely connecting to EC2 machine

feb25ct-key-pair.pem

vpc - subnet setting - more control over configuration, how can access what.

for connecting to this service from outside.
    have public IP,
    Security Group configuration

port 22
ssh - connect to machine
scp - copy files to machine

=> Launch instance

Once EC2 is setup
    Spring boot application

1. clean package -P dev

app.jar
2. Copy to AWS
scp
3. java -jar app.jar
nohup <command> &

EC2, securely connect
Install Java 17 on this machine

ssh -i <pemfile>

ssh -i D:\\ctdata\\aws\\feb25ct-key-pair.pem ec2-user@13.233.29.37

the machine which AWS manages, will have some security patches applied from time-to-time

sudo su

yum update
yum install java-17-amazon-corretto-17.0.10+7-1.amzn2.1.x86_64

java --version

===

Setup MySQL in RDS
Connect processing-service with RDS in Dev env
Deploy processing-service in EC2 & run the system.

RDS Creation:
Go to RDS
1. Create database => MySQL
2. mysql 8.x
3. Free tier
4. admin / <pwd>
5. public ip

Disaster Recovery

Highly Available

Java dev
- 3/4
    - frontend
    - Desinging

4-5-6
    - Infra

5+
    - Lead

====

Fully Setup RDS
1. We created RDS with MySQL
2. Connect to this RDS from local machine
3. DDL & DML

Java app side changes, to deploy in dev AWS env(ec2)

application-dev.properties
    update DB location to point to RDS

clean package -P dev
 jar

=> copy the jar in destination location

ssh -i D:\\ctdata\\aws\\feb25ct-key-pair.pem ec2-user@13.233.29.37

scp source destination

scp -i D:\\ctdata\\aws\\feb25ct-key-pair.pem ./payment-processing-service.jar ec2-user@13.233.29.37:/home/ec2-user

on ec2, give file permission
chmod 777 *

Security group.
- inbound rules
- allow 8082 to be connect from external places

RDS & connectivity

EC2 setup - processingservice (connects to RDS).
- Tested from local

AWS Secrets Manager

We need to build the system, that even if you want to break it, then you should not be able to break.

the sensitive data is env specific - service specific

1. Maintain sensitive data in Secrets Manager
2. Remove it form you application property files
3. You spring boot app should be able to connect & read the sensitive info from SecretsManager

dev env of processing service

- dev/payment-processing-service

- 2 sensitive information
- spring.datasource.password=cptraining
- stripe.endpointSecret=whsec_11c0af12865f8920...

dev/payment-processing-service

by default any other service is not able to connect to secrets manager - sensitive infor. So EC2 will not be able to connect.

We need to add additional configuration so that EC2 can connect to SM

aws secretsmanager get-secret-value --secret-id dev/payment-processing-service --region ap-south-1

1. Create an IAMRole
    EC2 => SecretsManager
    feb25-ec2-sm-iamrole

2. This IAMRole we will assign to our EC2 machine.

=> java spring boot changes to read from SM

---

1. Setup in SM with sensitive data
2. EC2 to connect to SM
3. Application side changes to read data from SM.
    remove the sensitive fields from property file

    spring.config.import=aws-secretsmanager:dev/payment-processing-service

    pom dependency

When trying to build
    clean package -P dev
    it runs testcases & because of @SpringBootTest, it activates spring containe (used for integration testing)
    when spring is activated it reads the property spring.config.import & tries to connect to AWS SM.
    Since local => SM configuration is not enabled, to build will fail.

=> use skip test, when building for envs.