## 1. Labo deploy aws

|  |  |
| --- | --- |
|  | sesion8/demo-aws |
| 1 | login  <https://aws.amazon.com/>   * ir a console * seleccionar region: us-east-2 |
| 2 | crear base de datos rds  seleccionar amazon rds  crear base de datos   * metodo: estandar create * engine: postgress * version: 9.5 * template: free tier * id: deploy-microservices-db * master username: master\_aws\_dev * master password: master\_aws\_dev   + confirm * connectivity   + vpc: default   + aditional config     - acceso publico: yes     - crear new security group       * name:postgress * aditional config   + initial database : eagle\_eye\_local |
| 3 | configurar usuario de base de datos  sudo apt-get install postgresql-client  psql \  --host=<database\_endpoint> \  --port=5432 \  --username=master\_aws\_dev \  --password \  --dbname=eagle\_eye\_local  CREATE USER postgres\_aws\_dev WITH ENCRYPTED PASSWORD 'postgres\_aws\_dev'; GRANT ALL PRIVILEGES ON DATABASE eagle\_eye\_local TO postgres\_aws\_dev; |
| 4 | configurar propiedades de base de datos en servicios   * url, username, password |
| 5 | crear usuario iam para conectarse a ECR  permisos   * + AmazonEC2ContainerRegistryFullAccess   + AmazonEC2ContainerServiceFullAccess |
| 6 | instalar aws cli  curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"  unzip awscliv2.zip  sudo ./aws/install |
| 7 | configurar aws cli con credenciales  aws configure  ingresar   * access key * secret key * region * output format: None |
| 8 | conectarse a ecr con aws cli  aws ecr get-login-password --region us-east-2 | docker login --username AWS --password-stdin **aws\_account\_id**.dkr.ecr.us-east-2.amazonaws.com |
| 9 | crear ecr repository , por cada servicio  aws ecr create-repository --repository-name licensing-service --image-scanning-configuration scanOnPush=true --region us-east-2  aws ecr create-repository --repository-name eurekasvr --image-scanning-configuration scanOnPush=true --region us-east-2 |
| 10 | preparar docker build:   * configurar poms con ecr uri * configurar docker-compose con ecr uri |
| 11 | build images  mvn clean package docker:build |
| 12 | push a ecr  docker push **aws\_account\_id**.dkr.ecr.us-east-2.amazonaws.com/licensing-service:chapter10  docker push **aws\_account\_id**.dkr.ecr.us-east-2.amazonaws.com/eurekasvr:chapter10 |
| 13 | crear key-pair para acceder a instancia ec2 |
| 14 | crear elastic container service cluster  buscar ecs   * ir menu lateral. clusters * select create cluster   + cluster template. ec2 linux networking   + next step * configure   + cluster name: deploy-microservices   + c2 instance type. t3.micro.   + key-pair     - name: deploy-microservices   + networking     - vpc: default-vpc     - subnets: seleccionar los subnets actuales     - security group: create new       * port 8080 |
| 15 | agregar acceso ssh   * buscar ec2 * select instancia * select security-group * seleccionar inbound * add rule   + ssh   + source: 0.0.0.0/0   chmod 400 deploy-microservices.pem  ssh -i "deploy-microservices.pem" ec2-user@<ec2-endpoint> |
| 16 | instalar amazon ecs-cli  sudo curl -o /usr/local/bin/ecs-cli https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-linux-amd64-latest  sudo chmod +x /usr/local/bin/ecs-cli  ecs-cli --version |
| 17 | configurar ecs-cli  AWS\_ACCESS\_KEY\_ID=<access\_key>  AWS\_SECRET\_ACCESS\_KEY=<secret\_key>  ecs-cli configure profile --profile-name microservices-profile --access-key $AWS\_ACCESS\_KEY\_ID --secret-key $AWS\_SECRET\_ACCESS\_KEY |
| 18 | conectar ecs-cli con cluster  ecs-cli configure --cluster deploy-microservices --default-launch-type EC2 --region us-east-2 --config-name deploy-microservices |
| 19 | desplegar contenedores  ecs-cli compose --file docker/common/docker-compose.yml up --cluster-config deploy-microservices --ecs-profile microservices-profile |
| 20 | verificar contenedores  ecs-cli ps --cluster-config deploy-microservices --ecs-profile microservices-profile |
| 21 | verificar logs  docker ps  docker logs -f <container-id> |
| 22 | acceder  http://endpoint-ec2:8080/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/licenses/ |
| 23 | modificar security-group rds   * buscar rds * select instance * ver security group rules * inbound * add rule   + postgress   + source anywhere |
| 24 | eliminar contenedores  ecs-cli compose --file docker/common/docker-compose.yml down --cluster-config deploy-microservices --ecs-profile microservices-profile |
| 25 | desplegar nuevamente  ecs-cli compose --file docker/common/docker-compose.yml up --cluster-config deploy-microservices --ecs-profile microservices-profile |
| 26 | limpiar recursos   * eliminar cluster * eliminar instancia rds * eliminar repositorios ecr |
| 27 | facturacion ver facturacion  ir billing managemebt console   * facturacion actual * free tier usage   + ver servicios y porcentaje de uso actual   configurar alerta al mail cuando se excede el 85% del limite de uso   * ir billing preferences * check recibir usage alerts * ingresar email * guardar   recibir alerts cuando excedas presupuesto   * ir budgets * crear budget * select cost budget * ingresar monto * configure alerts * enviar alerta cuando se exceda el 80% del presupuesto * crear |

# Labo messaging y cache

|  |  |
| --- | --- |
| 1 | * conexión con redis |
|  | desplegar   * mvn clean package docker:build && docker-compose -f docker/common/docker-compose.yml up |
| 2 | invocar  get http://localhost:5555/api/licensing/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/licenses/f3831f8c-c338-4ebe-a82a-e2fc1d1ff78a |
| 3 | construir producer en organization-service |
| 4 | construir consumer en licensing |
| 6 | invocar  put http://localhost:5555/api/organization/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/  body  {  "contactEmail": "mark.balster@custcrmco.com",  "contactName": "Mark Balster",  "contactPhone": "823-555-2222",  "id": "e254f8c-c442-4ebe-a82a-e2fc1d1ff78a",  "name": "customer-crm-co"  } |