**Talento Tech**

**Arquitectura en la nube**

**Nivel Explorador**

**Actividad 5**

Desarrollar una **API BASICA con un formulario de registro** que almacena los datos en **RDS MySQL**, se ejecuta desde una **instancia EC2 (Ubuntu Server)**, y utiliza **S3 para guardar archivos u objetos (por ejemplo, una foto de perfil)**, en una cuenta gratuita de AWS.

**📚 Bootcamp: Desarrollo de API de Registro con AWS RDS + EC2 + S3**

**Duración total**: 3 a 4 horas (puede dividirse en dos sesiones)

**🧩 PARTE 1: Introducción (15 minutos)**

**✅ Objetivo de la clase:**

* Crear una API REST con formulario de registro.
* Almacenar los datos del formulario en MySQL (RDS).
* Subir imágenes a S3.
* Desplegar API en EC2 Ubuntu Server.

**⚙️ PARTE 2: Preparar entorno local en VS Code (20 minutos)**

1. **Instalar dependencias necesarias en VS Code**:
   * Node.js (o Python, se usará Node.js en este ejemplo).
   * AWS CLI.
   * Git.
   * Extensión SSH para conectarse a la EC2.
2. **Conexión por SSH a la EC2**:

ssh -i "clave.pem" ubuntu@ec2-XX-XXX-XX-XXX.compute-1.amazonaws.com

1. **Actualizar e instalar Node.js y npm** en la EC2:

sudo apt update && sudo apt upgrade -y

sudo apt install nodejs npm -y

1. **Instalar MySQL client en EC2 para probar conexión a RDS**:

sudo apt install mysql-client -y

mysql -h <ENDPOINT\_RDS> -u admin -p

**🏗️ PARTE 3: Crear el backend API en Node.js (1 hora)**

**Estructura de carpetas:**

project/

│

├── app.js

├── config/

│ └── db.js

├── routes/

│ └── register.js

├── controllers/

│ └── userController.js

├── models/

│ └── user.js

└── uploads/

**1. Instalar librerías necesarias:**

npm init -y

npm install express mysql2 multer aws-sdk dotenv

**2. config/db.js: conexión a RDS**

const mysql = require('mysql2');

const connection = mysql.createConnection({

host: process.env.DB\_HOST,

user: 'admin',

password: process.env.DB\_PASSWORD,

database: 'formdb'

});

connection.connect(err => {

if (err) throw err;

console.log("Conectado a la base de datos RDS MySQL");

});

module.exports = connection;

**3. .env (crear este archivo)**

DB\_HOST=nombre-endpoint-rds.amazonaws.com

DB\_PASSWORD=clave

AWS\_ACCESS\_KEY\_ID=clave

AWS\_SECRET\_ACCESS\_KEY=clave

AWS\_REGION=us-east-1

S3\_BUCKET=my-form-bucket

**4. models/user.js**

const db = require('../config/db');

const saveUser = (userData, callback) => {

const sql = "INSERT INTO users (name, email, photo\_url) VALUES (?, ?, ?)";

db.query(sql, [userData.name, userData.email, userData.photo], callback);

};

module.exports = { saveUser };

**5. controllers/userController.js**

const AWS = require('aws-sdk');

const { saveUser } = require('../models/user');

const s3 = new AWS.S3();

exports.register = (req, res) => {

const { name, email } = req.body;

const file = req.file;

const params = {

Bucket: process.env.S3\_BUCKET,

Key: file.originalname,

Body: file.buffer,

ACL: 'public-read'

};

s3.upload(params, (err, data) => {

if (err) return res.status(500).send(err);

const userData = {

name,

email,

photo: data.Location

};

saveUser(userData, (err, result) => {

if (err) return res.status(500).send(err);

res.status(201).send({ message: 'Usuario registrado correctamente', data: userData });

});

});

};

**6. routes/register.js**

const express = require('express');

const multer = require('multer');

const upload = multer(); // memoria RAM

const { register } = require('../controllers/userController');

const router = express.Router();

router.post('/', upload.single('photo'), register);

module.exports = router;

**7. app.js**

require('dotenv').config();

const express = require('express');

const app = express();

const registerRoute = require('./routes/register');

app.use(express.json());

app.use('/register', registerRoute);

const PORT = 3000;

app.listen(PORT, () => {

console.log(`Servidor corriendo en puerto ${PORT}`);

});

**🧪 PARTE 4: Probar en EC2 (30 minutos)**

1. **Subir el código a EC2** (desde VS Code o con scp)

scp -i clave.pem -r ./project ubuntu@ip-ec2:/home/ubuntu/

1. **Instalar dependencias en EC2**:

cd project

npm install

1. **Crear tabla en RDS**

CREATE DATABASE formdb;

USE formdb;

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100),

email VARCHAR(100),

photo\_url TEXT

);

1. **Ejecutar el servidor**:

node app.js

**☁️ PARTE 5: Configurar Bucket S3 (20 minutos)**

1. Crear bucket en S3: my-form-bucket
2. Permitir **carga pública de archivos** o usar política personalizada (más seguro con IAM).
3. Habilitar CORS:

[

{

"AllowedHeaders": ["\*"],

"AllowedMethods": ["PUT", "POST", "GET"],

"AllowedOrigins": ["\*"],

"ExposeHeaders": []

}

]

**🧫 PARTE 6: Probar con Postman o HTML frontend (30 minutos)**

**HTML básico:**

<form action="http://IP\_PUBLICA\_EC2:3000/register" method="POST" enctype="multipart/form-data">

<input type="text" name="name" />

<input type="email" name="email" />

<input type="file" name="photo" />

<button type="submit">Enviar</button>

</form>

O probar en **Postman** con método POST, form-data:

* name: texto
* email: texto
* photo: archivo

**🔒 PARTE 7: Seguridad básica y buenas prácticas (15 minutos)**

* Usar HTTPS (certificado Let's Encrypt con NGINX).
* Crear roles IAM para acceso restringido a S3.
* Configurar grupo de seguridad para EC2 y RDS:
  + Puerto 3306 solo accesible desde IP EC2.
  + Puerto 3000 solo desde IP propia o frontend.

**✅ Conclusión y Tareas**

**✅ Evaluación sugerida:**

* Prueba de conexión a RDS.
* Prueba de carga de imágenes a S3.
* Registro exitoso desde frontend o Postman.

**📌 Tarea:**

* Agregar validaciones al formulario.
* Subir el código a GitHub.
* Mejorar diseño frontend.