--- Page 1 ---
AWS Services - Clear Explanation with Hands-On 1. Amazon EC2 (Elastic Compute Cloud) What it is: - A virtual machine on the cloud to run applications. When to Use: - When you need full control of OS, networking, storage. Hands-On Task: 1. Go to EC2 Console. 2. Launch a new instance with Amazon Linux 2. 3. Download the keypair (e.g., my-key.pem). 4. Connect via SSH: chmod 400 my-key.pem ssh -i "my-key.pem" ec2-user@ 5. Install Node.js: sudo yum update -y sudo yum install nodejs -y 6. Create and run a simple server: echo "console.log('Hello from EC2!')" > app.js node app.js 2. AWS Elastic Beanstalk What it is: - A service to deploy and manage applications easily without managing servers. When to Use: - When you want to deploy code quickly with auto-scaling and load balancing. Hands-On Task: 1. Install EB CLI: pip install awsebcli --upgrade --user export PATH="$PATH:$HOME/.local/bin" source ~/.bash_profile 2. Initialize Elastic Beanstalk: eb init 3. Create an environment: eb create my-env 4. Deploy your app: eb deploy eb open
--- Page 2 ---
AWS Services - Clear Explanation with Hands-On 3. Amazon ECS (Elastic Container Service) What it is: - Manages Docker containers running on EC2. When to Use: - You want to orchestrate containers with more control over infra. Hands-On Task: 1. Dockerize your app. 2. Push image to ECR: aws ecr create-repository --repository-name my-app # Tag and push Docker image 3. Create a Task Definition in ECS. 4. Run task in ECS cluster. 4. AWS Fargate What it is: - A serverless compute engine for containers. When to Use: - You want to run containers without managing EC2 instances. Hands-On Task: 1. Use same Docker image. 2. In ECS, choose Fargate launch type. 3. Define memory, CPU, and container image. 4. Deploy and access app via Load Balancer. 5. AWS Lambda What it is: - A serverless function that runs code on demand. When to Use: - Lightweight, event-driven tasks. Hands-On Task: 1. Go to AWS Lambda Console. 2. Create a function (e.g., Node.js runtime). 3. Write code in the inline editor: exports.handler = async (event) => {
--- Page 3 ---
AWS Services - Clear Explanation with Hands-On return { statusCode: 200, body: JSON.stringify('Hello from Lambda!'), }; }; 4. Test the function. 5. Add API Gateway to invoke it via HTTP. 6. AWS CodeCommit What it is: - A Git-based source control hosted by AWS. When to Use: - For private and secure version control. Hands-On Task: 1. Create a new repository in CodeCommit. 2. Configure Git credentials (via IAM). 3. Clone repository: git clone https://git-codecommit..amazonaws.com/v1/repos/my-repo 4. Push code: git add . git commit -m "Initial commit" git push origin main