📝 What is AWS User Data?

User Data in AWS is a script or set of instructions that you can provide to an **EC2 instance** at **launch time**. It's typically used to **automate initial setup and configuration** when the instance starts for the first time.

✅ **Common uses** include:

- Installing software (e.g., Apache, Docker)
- Updating packages
- Configuring environment variables
- Running startup scripts
- Downloading application code

📦 Where is User Data Applied?

- It is associated with an **EC2 instance**.
- Run **only during the first boot cycle by default**, unless configured otherwise.
- Supports **bash scripts**, **cloud-init**, or **PowerShell** (for Windows).

🛠️ Example: Linux Bash Script

Here's a simple **User Data** script that installs and starts Apache on an Amazon Linux EC2 instance:

```
#!/bin/bash
sudo yum update -y
sudo yum install -y httpd
sudo systemctl enable httpd
sudo systemctl start httpd
sudo echo "Hello from EC2 APP-1-Server" > /var/www/html/index.html
```

📌 How to use it:

- Paste it in the **"User Data"** field under **Advanced Details** during EC2 instance launch (in AWS Console).
- Or pass it via **CLI** or **SDK**.

⚙️ How to Provide User Data

✅ **AWS Console:**

- When launching an EC2 instance, go to:
  - **Step 3: Configure Instance Details**
  - Scroll to **Advanced Details**
  - Enter your script in the **User Data** text box

✅ **AWS CLI:**

```
aws ec2 run-instances \
  --image-id ami-12345678 \
```

```
  --instance-type t2.micro \
  --key-name MyKeyPair \
  --user-data file://user-data.sh
```

✅ **Terraform:**

```
resource "aws_instance" "web" {
  ami          = "ami-12345678"
  instance_type = "t2.micro"

  user_data = <<-EOF
            #!/bin/bash
            echo "Hello from Terraform" > /home/ec2-user/hello.txt
          EOF
}
```

🔄 Re-running User Data (Optional)

By default, **User Data runs only once**, but you can make it run every boot by modifying cloud-init settings:

Example (Amazon Linux):

Edit this file:

/etc/cloud/cloud.cfg

Change this line:

```
cloud_final_modules:
 - [scripts-user, always]
```

⚠️ Notes & Tips

- Scripts must begin with a **shebang (#!)** line.
- User Data has a **size limit of 16KB** when passed directly, more via multipart MIME.

Output of User Data execution is stored in:

/var/log/cloud-init-output.log

-

🧠 Summary

Feature	Detail
Purpose	Automate EC2 configuration at boot
Scripting formats	Bash (Linux), PowerShell (Windows), cloud-init
Execution timing	Default: once at first boot
Max size	16KB (direct input)
Storage	Log: /var/log/cloud-init-output.log

Sure — here are direct examples:

🟦 Application Load Balancer (ALB) – Examples

1. **E-commerce website**

- Routes `/shop` to frontend, `/api` to backend service.

2. **Microservices on ECS/EKS**
   - Routes traffic to containers based on path or hostname.
3. **Multi-tenant SaaS**
   - `tenant1.example.com` → Tenant 1's stack, `tenant2.example.com` → Tenant 2's stack.
4. **OAuth-secured web app**
   - Uses ALB with Cognito for user login before reaching app.
5. **Single-page app (SPA)**
   - Serves the frontend via ALB with HTTPS and custom error pages.

🟨 Network Load Balancer (NLB) – Examples

1. **High-performance game server**
   - Routes TCP traffic with ultra-low latency.
2. **IoT data ingestion**
   - Handles millions of UDP connections from IoT devices.
3. **SMTP mail relay**
   - Routes TCP port 25 traffic to backend mail servers.
4. **VPN gateway**

    - Passes IPSec VPN traffic directly to backend instances.

5. **Financial trading platform**
    - Requires extremely low latency TCP load balancing.
