Step 1: Plan Your Architecture

Before creating resources, plan:

- **VPC**: One VPC with **public and private subnets**.
- **EC2**: Web server in **public subnet**.
- **RDS**: Database in **private subnet**.
- **EBS**: Attached to EC2 for persistent WordPress storage.
- **S3**: For media uploads.
- **IAM Role**: Give EC2 permission to access S3.
- **Security Groups**: EC2 allows HTTP/HTTPS, RDS allows only EC2 to connect.

Step 2: Create a VPC

1. Go to **VPC Console → Create VPC**.
    - Name: `WordPressVPC`
    - CIDR: `10.0.0.0/16`
2. Create **subnets**:
    - Public subnet (EC2): `10.0.1.0/24`
    - Private subnet (RDS): `10.0.2.0/24`
3. Create **Internet Gateway** → Attach to VPC.
4. Create **Route Table** for public subnet → Route 0.0.0.0/0 → Internet Gateway.
5. Associate public subnet with this route table.

Step 3: Security Groups

1. **EC2 Security Group**:
    - Inbound: HTTP (80), HTTPS (443), SSH (22) from your IP.
    - Outbound: All traffic.
2. **RDS Security Group**:
    - Inbound: MySQL/Aurora (3306) from EC2 Security Group only.
    - Outbound: All traffic.

Step 4: Create IAM Role for EC2

1. Go to **IAM → Roles → Create Role**
2. Choose **AWS Service → EC2**
3. Attach policy: `AmazonS3FullAccess` (or restricted bucket access policy)

4. Name it: `EC2S3AccessRole`

Step 5: Create an RDS Database

1. Go to **RDS → Create Database**
2. Engine: **MySQL** (or MariaDB)
3. DB instance class: e.g., `db.t3.micro` (for testing)
4. Storage: 20 GB (auto-scalable optional)
5. **Connectivity**:
   - VPC: `WordPressVPC`
   - Subnet group: private subnet
   - Public accessibility: **No**
6. Security group: RDS SG created earlier
7. Database credentials: Note down `DB_USERNAME` and `DB_PASSWORD`.

Step 6: Launch EC2 Instance

1. Go to **EC2 → Launch Instance**
2. AMI: Amazon Linux 2 or Ubuntu
3. Instance type: t2.micro (for testing)
4. Network: `WordPressVPC` → Subnet: Public subnet
5. IAM Role: Attach `EC2S3AccessRole`
6. Storage: Add **EBS volume** (e.g., 20 GB)
7. Security Group: Use EC2 SG

Step 7: Install WordPress via User Data

1. During EC2 launch, in **Advanced Details → User Data**, paste a script like:

```
#!/bin/bash
yum update -y
yum install -y httpd php php-mysqlnd wget unzip
systemctl start httpd
systemctl enable httpd

# Download WordPress
cd /var/www/html
wget https://wordpress.org/latest.zip
unzip latest.zip
cp -r wordpress/* .
```

```
rm -rf wordpress latest.zip

# Set permissions
chown -R apache:apache /var/www/html

# Configure wp-config.php
DB_NAME="your_db_name"
DB_USER="your_db_user"
DB_PASS="your_db_password"
DB_HOST="your_rds_endpoint"

cp wp-config-sample.php wp-config.php
sed -i "s/database_name_here/$DB_NAME/" wp-config.php
sed -i "s/username_here/$DB_USER/" wp-config.php
sed -i "s/password_here/$DB_PASS/" wp-config.php
sed -i "s/localhost/$DB_HOST/" wp-config.php

# Install AWS CLI to sync S3
yum install -y awscli
mkdir /var/www/html/wp-content/uploads
aws s3 sync s3://your-bucket-name /var/www/html/wp-content/uploads
```

2. Launch the instance → It will automatically install WordPress and connect to RDS.

Step 8: Configure S3 for Media Uploads

1. Create S3 bucket: `wordpress-media-bucket`
2. Set bucket policy if you want public access (optional).
3. EC2 IAM role already has access → User Data script syncs S3 with WordPress uploads folder.
4. You can also use plugins like **WP Offload Media** for live sync.

Step 9: Test the WordPress Site

1. Go to `http://<EC2_public_IP>` → WordPress setup screen appears.
2. Complete the setup: site title, admin user, password.
3. Upload media → check if it syncs with S3 (or via plugin).

Step 10: Optional Enhancements

- Enable **HTTPS** with Let's Encrypt
- Enable **backups** for RDS and S3

- Auto-scaling for EC2
- CloudFront CDN for faster media delivery