To deploy to AWS, you can use the AWS Management Console, AWS CloudFormation, AWS Cloud Development Kit (AWS CDK), or AWS CodeDeploy.

1. Create an instance: Create an EC2 instance or provision an instance
2. Configure security: Set up security groups to act as a virtual firewall
3. Prepare your application: Create an application revision, bundle it, and upload it to Amazon S3
4. Deploy your application: Use the AWS Management Console, AWS CloudFormation, AWS CDK, or AWS CodeDeploy
5. Test your application: Open a web browser and test that your code is working
6. Clean up: Remove any resources you no longer need

https://www.youtube.com/watch?v=goiW0g7A0WE&t=0

Step 1: Set Up an AWS Account

1. Create an AWS Account:

- If you don't already have one, sign up at [AWS](https://aws.amazon.com/).

- Complete the verification process by providing billing information.

- You'll have access to the AWS Management Console, where you can manage all your AWS resources.

Step 2: Plan Your VPC Setup

Before deploying your EC2 instances, you'll need to set up a VPC to host them. A VPC allows you to create an isolated network environment in the AWS cloud.

1. Log in to the AWS Management Console.

2. Go to the VPC Dashboard:
    - a. In the AWS Console, search for "VPC" and open the VPC dashboard.
3. Create a VPC:
    - b. Navigate to the “VPC Wizard” or click "Create VPC."
    - c. Configure the VPC Settings:
    - d. CIDR Block (e.g., `10.0.0.0/16`).
    - e. Name the VPC.
    - f. Create Subnets:
    - g. Create a Public Subnet (e.g., `10.0.1.0/24`) for internet-accessible resources.
    - h. Create a Private Subnet (e.g., `10.0.2.0/24`) for internal resources.
4. Set Up Route Tables:
    - i. Create a route table for the public subnet to route traffic to the internet using an Internet Gateway (IGW).
    - j. For the private subnet, create a route table with access to the internet via a NAT Gateway.
5. Create Internet Gateway:
    - k. Attach an Internet Gateway to the public subnet.
6. Security Groups and Network ACLs:
    - l. Configure Security Groups to control traffic to and from EC2 instances.
    - m. Set up Network ACLs for further access control between subnets.

Step 3: Create and Configure EC2 Instances

1. Launch an EC2 Instance:
    - Go to the EC2 Dashboard and click “Launch Instance.”
    - Choose an AMI: Select the Amazon Machine Image (AMI) of your choice (e.g., Ubuntu, Amazon Linux).
    - Choose an Instance Type: Select an EC2 instance type based on your requirements (e.g., `t2.micro` for a small test instance).
    - Configure Instance Details:
    - Select the VPC and Subnet where the EC2 instance will be launched.
    - Enable "Auto-assign Public IP" for public-facing instances.

- Add Storage: Configure your instance’s storage requirements.
- Add Tags: Add tags to identify the instance (optional).
- Configure Security Group:
- Select an existing Security Group or create a new one.
- Ensure the necessary ports are open (e.g., port 22 for SSH access or port 80 for HTTP).
- Launch the Instance: Choose an existing key pair or create a new one to securely access the instance via SSH.

2. Access Your EC2 Instance:
    - Once the instance is running, use the public IP to SSH into it (for Linux instances) or RDP (for Windows instances).
    - Example SSH Command (Linux):

      ```bash
      ssh -i "your-key.pem" ec2-user@<your-instance-public-ip>
      ```

Step 4: Configure Your Application on EC2

Once your EC2 instance is up and running, you can install and configure the necessary software for your application.

1. Install Software Packages: SSH into the EC2 instance and install the required software (e.g., web servers like Apache, Nginx, or database services).

   - Example: For a basic web server (Apache), you can run:

```bash
sudo yum update -y

sudo yum install httpd -y

sudo systemctl start httpd

sudo systemctl enable httpd
```

2. Deploy Application: Upload your application files (e.g., HTML, PHP, Python) to the instance and configure it to run.

Step 5: Set Up Load Balancer (Optional)

For high availability, you might need to set up a load balancer to distribute traffic across multiple EC2 instances.

1. Navigate to the ELB Dashboard:
   Go to the EC2 dashboard and select "Load Balancers" under "Load Balancing."
2. Create a Load Balancer:
   Choose an Application Load Balancer (ALB) for HTTP/HTTPS traffic or a Classic Load Balancer (CLB) for older architectures.

   - Configure listeners (e.g., port 80 or 443 for HTTPS).

   - Register your EC2 instances with the load balancer.

Step 6: Set Up Auto Scaling (Optional)

For scalability, use Auto Scaling to automatically add or remove EC2 instances based on traffic demand.

1. Create an Auto Scaling Group:

   - Go to the EC2 Dashboard and click on "Auto Scaling Groups."

- Create a Launch Configuration based on your EC2 instance.

- Set the desired instance count and configure scaling policies.

2. Configure Auto Scaling Policies:

   - Set up scaling rules (e.g., add an instance when CPU utilization > 80% and remove when it’s < 20%).

Step 7: Monitor the Deployment

Use AWS services to monitor the health and performance of your EC2 instances and VPC infrastructure.

1. Amazon CloudWatch:

   Use CloudWatch to monitor instance performance (e.g., CPU usage, disk I/O, network traffic).

2. AWS CloudTrail:

   - Enable CloudTrail to log API calls and track changes made to resources for auditing.

Step 8: Set Up DNS (Optional)

If you want to assign a custom domain name to your application, use Amazon Route 53.

1. Create a Hosted Zone:

- Navigate to the Route 53 console and create a new hosted zone for your domain.

2. Set Up DNS Records:

   - Create A or CNAME records to point your domain name to the public IP or Load Balancer DNS name.

Step 9: Implement Security Best Practices

Make sure your deployment is secure by following these security practices:

1. IAM Roles and Policies:

   Use IAM roles to assign permissions to your EC2 instances instead of using access keys.

2. Security Groups:

   Properly configure security groups to allow only necessary inbound and outbound traffic.

3. Encryption:

   Use SSL certificates (for HTTPS) and encrypt sensitive data in your EC2 instances and S3 buckets.

Step 10: Backup and Maintain Your Deployment

1. Backups:

   Set up regular backups of your EC2 instance using Amazon EC2 snapshots or use AWS Backup service for automated backups.

2. Update and Patch:

Regularly update your EC2 instances with security patches and new versions of the application.

Step 11: Scale and Optimize Your Infrastructure

As your application grows, you might need to scale your resources or optimize costs.

1. Vertical Scaling:

   Increase EC2 instance size (CPU, memory) to handle more load.
2. Horizontal Scaling:

   Use Auto Scaling Groups and Load Balancers to handle increasing traffic.