# AWS Cloudformation

AWS Cloudformation is an infrastructure as Code (IaC) servic that let you define, provision, and manage Aws resources in a declaratiu, templates-based formats.

## Template Example :-

```
Resources:
  Simple EC2 Instances:
    Type: "AWS::EC2::Instance"

    Properties:
      InstanceType : t2.micro
      ImageId: ami-02 ___ AMI ID your region
      Tags:
        -Key: Name
         Value: My Intance.
```

## Steps for Cloud formation

1. Search → Cloud formation
2. Create Stack → choose template → upload template.
3. Edit in Infrastructure Composer
4. Then Next.

5. Give Stack a name → next.
6. Add IAM role & default
7. Next and preview & role check.
8. Confirm & creat Stack.
9. Also update & replace the existing one.

## Key Concept

1. **Template** :- ≐ Written in Yaml or Json
≐ Describe what resources you want (EC2, S3, IAM) etc.

2. **Stack** :- A collection of AWS resources created and managed together using a template.

3. **Resources** :- Actual AWS services you define inside the template (e.g. AWS::S3::bucket).

4. **Parameters** :- Dynamic inputs to customize your templates (like EC2 instance type or VPC id).

5. **Outputs** :- Return values after the stack is created. (like IP address bucket name).

6. **Mapping** :- Static variables like regions or AMI Ids, used within the template.

7. Conditions :- logic to include / exclude resources based on - parameter value.

8. Transform :- Supports macros like AWS::Include or Serverless Application model.

## Use-Cases → key points

1. Consistency :- Define infrastructure in code, consistently like if make EC2 instance again & again so use this cloud formation template.

2. Automation :- Reduce manual work and make the work Automated as like want to make EC2 instance stop / pause at any event trigger using templates.

3. Repeatable :- Replicate environments easily → environment means all the template data replicate in any other code or place.

# Some important points about Cloudformation

1. Infrastructure as Code (IaC) tool :- Automates the creation, management, and updating of AWS infrastructure

2. Declarative language :- Users define desired end states for resources and cloud formation handles provisioning.

3. Template Based :- Use Yaml or Json templates to specify AWS resources & configuration.

4. AWS Native :- Exclusively supports AWS resources, fully integrated with AWS services.

5. State management :- manages state immediately, eliminating the need for separate state files.

6. Cost free tool :- Cloud formation itself is free; you only pay for the AWS resources created.

7. Drift detection :- Identifies & reports on resources changes made outside of cloud formation to ensure resources stay

aligned with the templates

8. Stack and Stack Set + Organise resources in stacks for easier management & allows for multi-accounts & region deployements with Stack sets.

## Some Important Usecases

1. Create EC2 Instance with Security Group & Elastic IP.
2. Provision S3 Buckets with Http Endpoints
3. Setup VPCs with Subnet & Route tables.
4. Crearete RDS database with Automate backup.
5. deploy Lambda function with API Gateway Intrgration
6. Set up Elastic Load Balancers and Auto Scaling for Web Application
7. deploy IAM Roles, policies, & User Access management.

## #Automation of template Using CLI

```
aws cloudformation create-stack --stack-name
MyStack --template-body
file://path/to/template-yml
```