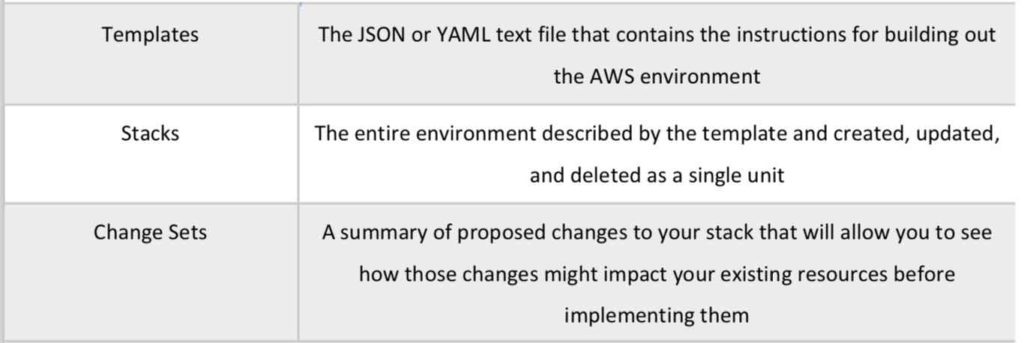
**CloudFormation**

* CF is a service that gives developers and businesses an easy way to create a collection of related AWS resources and provision them in an orderly and predictable fashion
* CF provides a common language for you to describe and provision all the infrastructure resources in your cloud environment
* CF can be used to provision a broad range of AWS resources
* Think of CF ad seploying infrastructure as code
* Elastic Beanstalk is more focused on deploying applications on Ec2(Paas)
* CF can deploy Elastic Beanstalk-hosted applications, however the reverse is not possible
* Logical IDs are used to reference resources within the template
* Physical IDs identify resources outside of CF templates, but only after the resources have been created
* Concept of templates, stacks and change sets:
* 
* Templates:
  + Architectural designs
  + Create, update and delete templates
  + Written in JSON or YAML
  + CF determines the order of provisioning
  + Don’t need to worry about dependencies
  + Modifies and updates templated in a controlled way(version control)
  + Designer allows you to visualize using a drag and drop interface
* Stacks:
  + Deployed resources based on templates
  + Create, update and delete stacks using templates
  + Deployed through Management Console, CLI or APIs
* Template elements:
  + Mandatory
    - File format and version
    - List of resources and associated configuration values
  + Not mandatory
    - Template parameters(limited to 60)
    - Output values(limited to 60)
    - List of data tables
* Puppet and Chef integration is supported
* Can use bootstrap scripts
* Can define delietion policies
* Provides WaitCondition function
* Can create roles in IAM
* VPCs can be created and customized
* VPC peering in the same AWS account can be performed
* R53 is supported
* Stack creation errors:
  + Automatic rollback on error is enabled by default
  + You will be charged for resources provisioned even if there is an error
* Updating stacks
  + CF provides two methods for updating stacks: direct update or creating and executing change sets
  + When you directly update a stack, you submit changes and CF immediately depolys them
  + Use direct updates when you want to quickly deploy your updates
  + With change sets, you can preview the changes CF will make to your stack, and then decide whether to apply those changes
* StackSets:
  + CF StackSets extends the functionality of stacks by enabling you to create, update, or delete stacks across multiple accounts and regions with a single operation
  + Using an administrator account, you define and manage an AWS CF template, and use the template as the basis for provisioning stacks into selected target accounts across specified regions
  + An administrator account is the AWS account in which you create stack sets
  + A stack set is managed by signing into the AWS administrator account in which it was created
  + A target account is the account into which you create, update or delete one or more of the stacks in your stack set
* Best Practices:
  + AWS provides Python ‘helper scripts’ which can help you install software and start services on your EC2 instances
  + Use CF to make changes to your landscape rather than going direclty into the resources
  + Make use of Change Sets to identify potential trouble spots in your updates
  + Use Stack Policies to explicity protect sensitive portions of your stack
  + Use a version control system such as CodeCommit or GitHub to track changes to templates
* Charges
  + There are no additional charges for CF
  + You pay for AWS resources(such as Ec2 instances, ELBs) creatged using CF
  + You pay only for what you use, as you use it etc