What is CloudFormation?

- CloudFormation (cfn) is an AWS language to create **infrastructure as code** (IAC).
  - **Infrastructure as code**: Allows one to specify resources that can be created on the cloud, and version control them
- When you create a CloudFormation "*template*", you deploy it as a "*stack*"
- Can be written in YAML or JSON

YAML/YML

- Main page: https://yaml.org/
- General mappings use a colon and a space (key:value)
- Lists begins with a dash and then a space

```
MyItems:
- EntryOne
- EntryTwo
```

- An alternative way of referencing a list of sequences is through square brackets "[]" separated by commas
  `MyItems: [EntryOne, EntryTwo]`
- `#` Begins a comment
- Pipe "|" symbol in YML: What follows is a multi-line scalar value
- Greater than ">" symbol in YML: Gets rid of newlines in what follows

```
include_newlines: |
        exactly as you see
        will appear these three
        lines of poetry

fold_newlines: >
        this is really a
```

```
        single line of text
        despite appearances
```

Sections

a. Format Version –version of AWS CloudFormation template; currently there is only one version: "2010-09-09"
b. Description – documentation on the template
c. Metadata -- stores objects in JSON or YML
d. Parameters – inputs passed to stack. Can be referenced in resources/outputs. String parameters can be referenced using ${}
e. Conditions – define conditions under which a resource can be created or configured
f. Mappings – commonly used to specify Amazon Resource Names (ARNs) or AMIs in different regions
g. Transform – specifies macros that CloudFormation uses to process your template. Can be used to translate different types of syntax into CloudFormation.
h. Resources – define AWS resources and their properties. The only required section.
i. Outputs – Can be viewed in the Console, or imported into another stack.

Intrinsic Functions

There are a number of intrinsic functions for CloudFormation. Short-hand version uses an explanation point before the function name

1. **Ref** - returns the value of the parameter or resource.

```
!Ref MyResource
```

2. **Base64** - returns the Base64 representation of the input string. (Commonly used with !Sub and the YAML pipe character to provide several lines user data)
3. **Sub** - substitutes variables in an input string with values you specify.
   Example #2:

```
UserData:
    Fn::Base64:
        !Sub |
            #!/bin/bash -xe
            # other commands
```

4. **Join** - Appends a set of values into a single value, separated by the specified delimiter
5. **GetAtt** - Returns the value of an attribute from a resource in a template (some resources have attributes)

```
!Join [ '', [ 'http://', !GetAtt MyELB.DNSName ] ]
```