Environment:

An environment is a way to map an organization’s real-life workflow to what can be configured and managed when using Chef server. Every organization begins with a single environment called the `_default` environment, which cannot be modified (or deleted). Additional environments can be created to reflect each organization’s patterns and workflow. For example, creating `production`, `staging`, `testing`, and `development` environments. Generally, an environment is also associated with one (or more) cookbook versions.

The `knife environment` subcommand is used to manage environments within a single organization on the Chef server.

The _default Environment

Every organization must have at least one environment. Every organization starts out with a single environment that is named `_default`, which ensures that at least one environment is always available to the Chef server. The`_default` environment cannot be modified in any way. Nodes, roles, run-lists, cookbooks (and cookbook versions), and attributes specific to an organization can only be associated with a custom environment.

Environment Formats

Environment data may be stored in two formats:

- As Ruby (i.e. a file that ends with `.rb`); this format is not available when running the chef-client in local mode
- As JSON (i.e. a file that ends with `.json`)

Refer: https://docs.chef.io/environments.html

Example of Environment:

1. Navigate to root directory of chef-repo
2. Create a directory called environments
3. Create a file called as dev.rb,prod.rb,test.rb

   ```
   name "dev"
   description "This is the development version"
   cookbook "nginx", "= 0.1.1"
   override_attributes(
   {
   ```

```
"webserver" => {
            "name" => "apache2"
         }

}
)
```

4. Upload environment using knife environment from file <filename>
5. Environment can be applied to file by changing in management console or navigating to /etc/chef in node and adding a line environment “dev” to client.rb file

Roles:

A role is a way to define certain patterns and processes that exist across nodes in an organization as belonging to a single job function. Each role consists of zero (or more) attributes and a run-list. Each node can have zero (or more) roles assigned to it. When a role is run against a node, the configuration details of that node are compared against the attributes of the role, and then the contents of that role’s run-list are applied to the node’s configuration details. When a chef-client runs, it merges its own attributes and run-lists with those contained within each assigned role.