**EXPERIMENT NO: 11**

**DEVOPS**

**Aim:** To learn software configuration management and provisioning using Puppet blocks (Manifest, Modules, Classes, Function).

**LO:6** – Synthesis software configuration and provisioning using Ansible.

**Theory:**

**Puppet Manifest:**

* In puppet, all the programs are written in Ruby programming language and added with an extension of .pp is known as manifests. The full form of .pp is the puppet program.
* Manifest files are puppet programs. This is used to manage the target host system. All the puppet programs follow the puppet coding style.
* We can use a set of different kinds of resources in any manifest, which is grouped by definition and class.
* Puppet manifest also supports the conditional statement. The default manifest file is available in the /etc/puppet/manifests/site.pp location.63.9M

**Manifest Components**

Puppet manifest has the following components:

* **Files:** Files are the plain text files that can be directly deployed on your puppet clients. Such as yum.conf, httpd.con, etc.
* **Resources:** Resources are the elements that we need to evaluate or change. Resources can be packages, files, etc.
* **Templates:** This is used to create configuration files on nodes and which we can reuse later.
* **Nodes:** Block of code where all the information and definition related to the client are defined here.
* **Classes:** Classes are used to group different types of resources.

**Puppet Modules:**

* Puppet Module is a collection of files, classes, templates, and resources. Each module handles a specific task in your infrastructure, such as installing and configuring a piece of software.
* Since modules allow you to divide your code into multiple manifests, it is very helpful in organizing your puppet code. Modules are the reusable and shareable units in the [puppet](https://www.javatpoint.com/puppet).
* Modules must be installed in the puppet modulepath. And the modulepath is /etc/puppet/modules directory.

**Puppet Classes:**

* Puppet classes are the set of puppet resources that are grouped together as a single unit. Classes are used to model the fundamental aspects of the node. Puppet uses classes to make the structure reusable and organized. Classes can only be evaluated once per node.
* Classes are described within the manifest file, located inside [Puppet modules](https://www.javatpoint.com/puppet-modules). The main reason for using a class is to decrease the duplication of the same code inside any manifest file or other [puppet](https://www.javatpoint.com/puppet) code.

**Puppet Functions:**

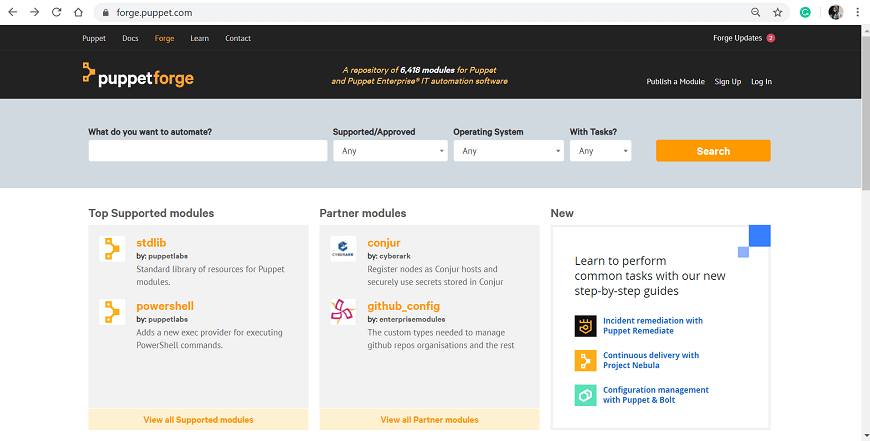
* As we know, Puppet uses Ruby programming language, and like other programming languages, Ruby also supports the function. We can write the functions in Ruby language and can be distributed in puppet modules. Puppet provides two different types of functions:
* **statement:** This type of function did not return any value and used to perform standalone tasks. It can be used to import Puppet modules in the new manifest file.
* **rvalue:** In Puppet, if you want to define a function with their return type, then you can use rvalue functions. rvalue can only be used when the statement needs a value, like a case statement or assignment. A function can only take two parameters as an argument.

**Steps and Commands:**

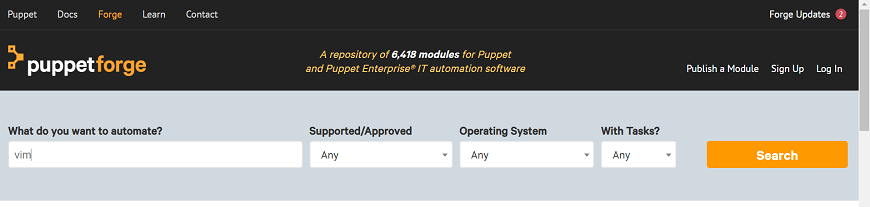
**Modules:**

**Step 1:** [Click here](https://forge.puppet.com/) to open the official Puppet Forge page.

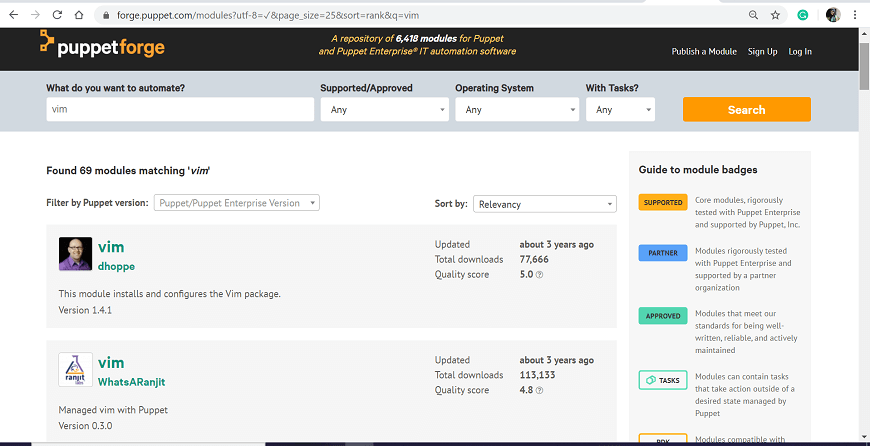
When you click on the link, it will display the following page:



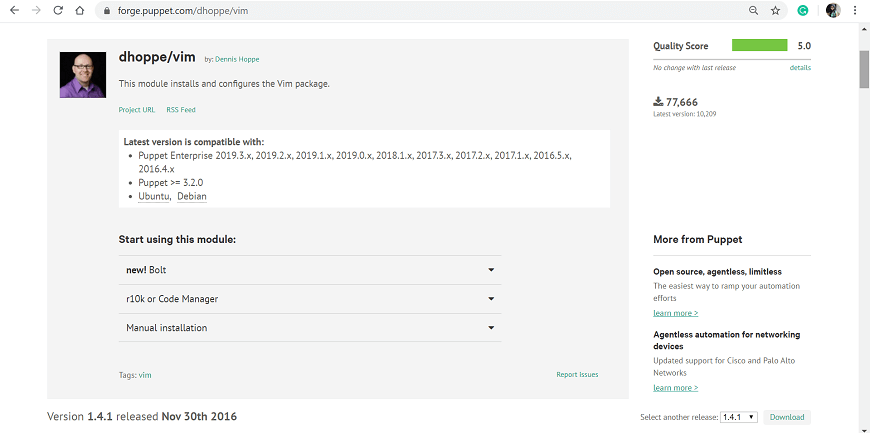
**Step 2:** In the search bar, enter the name of the module that you want to download. Here, we are going to download vim module:



**Step 3:** When you click on the Search button, it will display multiple results. Choose the appropriate one. To decide your selection, you can click on the individual module to see complete details. In our case, we will use dhoppe vim.

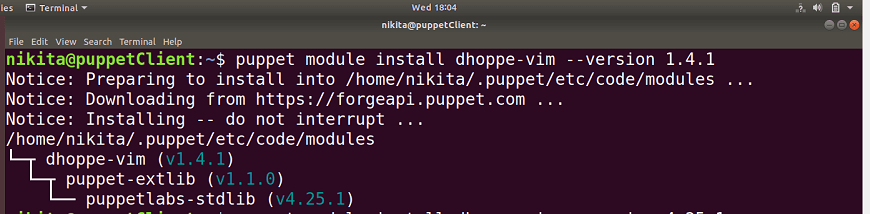


**Step 4:** To download the module, click on the download button from the right side of the page, and we will get the module in tarball format.



**Step 5:** Once the download finish, execute the following command to install a module from the tarball:

* puppet module install /path/dhoppe-vim-1.4.1.tar.gz
* In the above command, 'path' is the path of the directory where your tarball is saved.We can also install the puppet modules online.To download and install the module from the puppet module tool, execute the following command:
* puppet module install dhoppe-vim --version 1.4.1



**Syntax:**

**class** my\_class {

  ... puppet code ...

}

include my\_class

**class** unix {

   file {

      '/etc/passwd':

      owner => 'superuser',

      group => 'superuser',

      mode => 644;

      '/etc/shadow':

      owner => 'nikita',

      group => 'nikita',

      mode => 440;

   }

}

Let's see another simple example which is similar to the above example:

**class** unix {

file {

'/etc/passwd':

owner => 'superuser',

group => 'superuser',

mode => 644;

}

file {'/etc/shadow':

owner => 'nikita',

group => 'nikita',

mode => 440;

}

}

Parameterized Class : Parameters are used to allow a class to request external data. If a class has to configure itself to data other than facts, the data will typically be inserted by a parameter into the class.

Let's see one example:

**class** windows\_ntp (

  String $server = 'time.windows.com',

) {

  registry::value { 'NtpServer':

    key  => 'HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters',

    data => "${server},0x9",

  }

  service { 'w32time':

    ensure => running,

    enable => **true**,

  }

}

**Conclusion:**