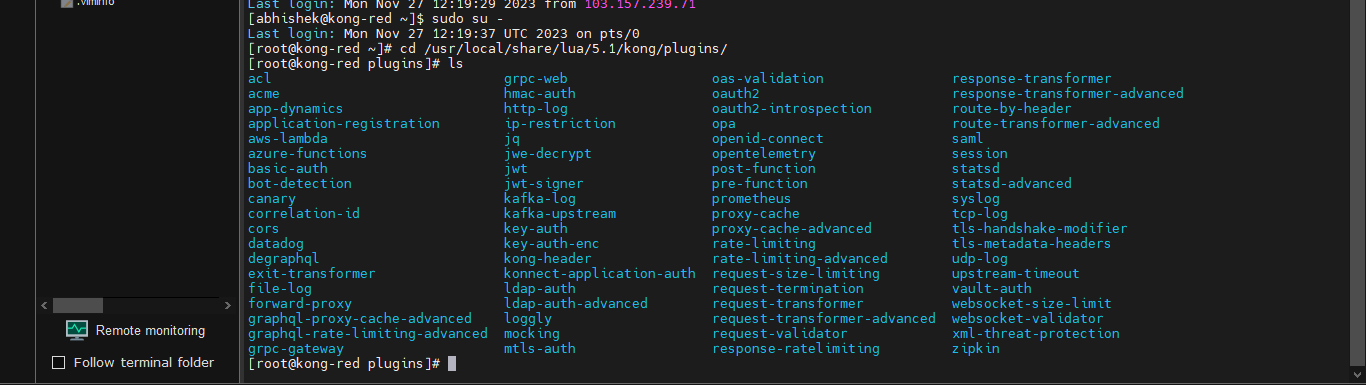
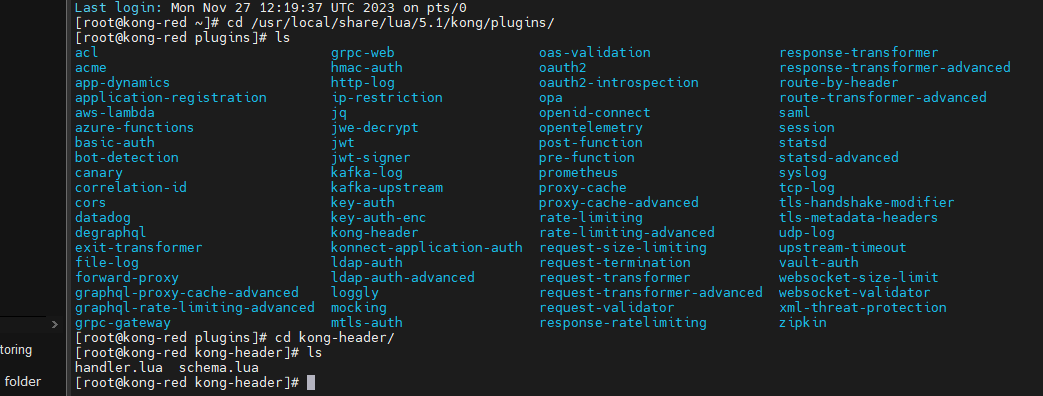
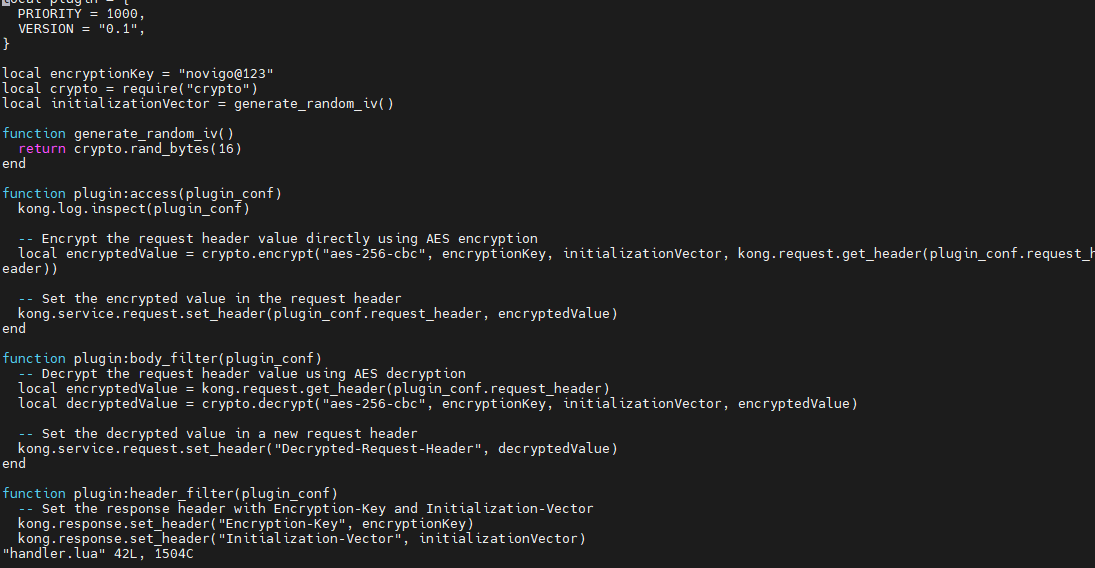
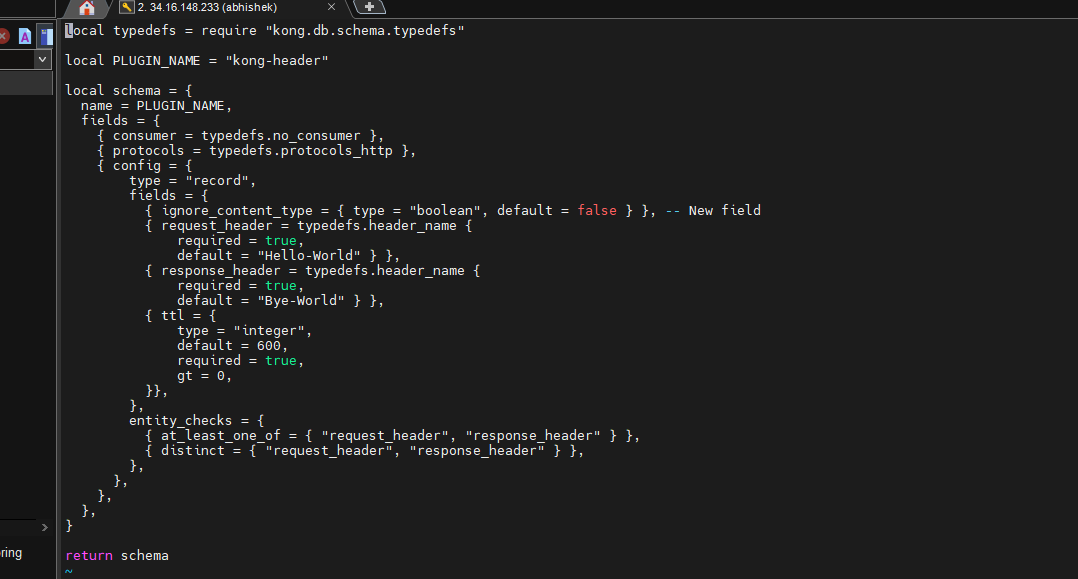
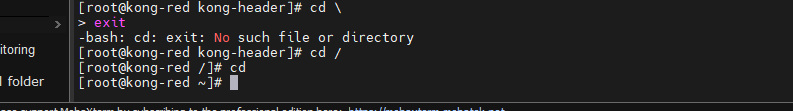
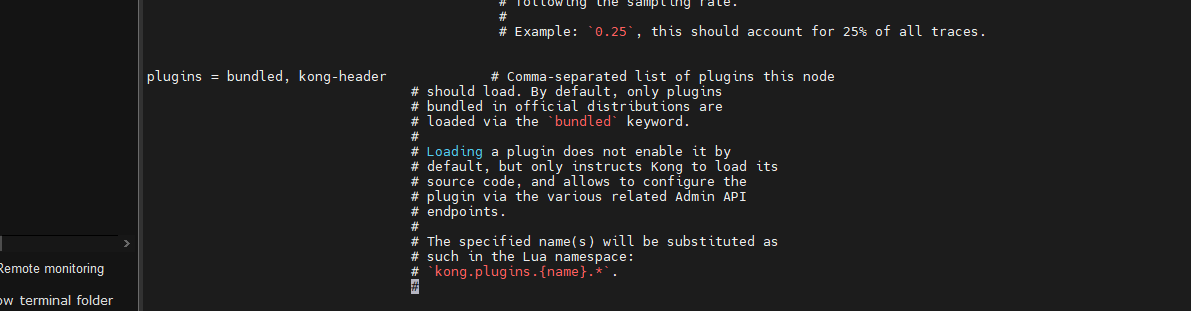
**Custom Plugin Environment Setup:**

* You can do the setup in docker-compose and docker also follows similar steps as mentioned below
* For now i will explain how to do the setup the custom plugin development in redhat linux and same steps follows for all.
* To use linux u need to become the root user so always use command as
* This one sudo su -
* Then go to the directory as mentioned below
* Cd /usr/local/share/lua/5.1/kong/plugins
* 
* Do cd plugins
* Then create a folder inside plugins example as kong-header
* Then inside kong-header create two files handler.lua and schema.lua
* In schema.lua u need to add frontend logic and handler.lua you need to add backend logic.
* 
* Once you come inside cd kong-header
* Then to create handler and schema files you can run the command as as vi handler.lua
* 
* Then you can copy the logic into handler file
* Then run the command as vi schema.lua it will create schema.lua file then u can copy the frontend logic command
* 
* After doing this steps you need to add the folder name where you created schema and handler file into kong.conf file
* To come out of a directory u can do like this
* 
* Next do cd /etc/kong
* 
* Do vi kong.conf
* Then search /plugins and then remove “#” and then add the plugin folder name
* 
* Then run the command as “kong restart”
* These steps we need follow for custom plugin development
* Here below you have sample schema.lua and handler.lua which can be used for other plugins setup where you can modify the below code for other plugin development
* First one is handler.lua

local plugin = {

PRIORITY = 1000,

VERSION = "0.1",

}

function plugin:access(plugin\_conf)

kong.log.inspect(plugin\_conf)

kong.service.request.set\_header(plugin\_conf.request\_header, "this is on a request")

end

function plugin:header\_filter(plugin\_conf)

kong.response.set\_header(plugin\_conf.response\_header, "this is on the response")

end

return plugin

* Here below u have schema.lua file sample code is given below

local typedefs = require "kong.db.schema.typedefs"

local PLUGIN\_NAME = "kong-header"

local schema = {

name = PLUGIN\_NAME,

fields = {

{ consumer = typedefs.no\_consumer },

{ protocols = typedefs.protocols\_http },

{ config = {

type = "record",

fields = {

{ ignore\_content\_type = { type = "boolean", default = false } }, -- New field

{ request\_header = typedefs.header\_name {

required = true,

default = "Hello-World" } },

{ response\_header = typedefs.header\_name {

required = true,

default = "Bye-World" } },

{ ttl = {

type = "integer",

default = 600,

required = true,

gt = 0,

}},

},

entity\_checks = {

{ at\_least\_one\_of = { "request\_header", "response\_header" } },

{ distinct = { "request\_header", "response\_header" } },

},

},

},

},

}

return schema

* Here is the complete setup