**Magento Edition**

1. Community
2. Enterprises

**Magneto 2 Development Environment Setup**

Step No. 1 create account on magento marketplace

Step No. 2 Create a vagrant box for Ubuntu

Step No. 3 System Requirements

1. Ubuntu Operating System
2. Memory RAM minimum 2GB& swap file of 2GB
3. Latest Composer
4. Web Server (Apache 2.4 or Nginx 1.x )
5. Mysql 5.6 or 5.7
6. PHP 7.1.3 or 7.2.0varified with PHP 7.2.11
7. PHP Extensions
   * 1. ext-bcmath
     2. ext-ctype
     3. ext-curl
     4. ext-dom
     5. ext-gd
     6. ext-hash
     7. ext-iconv
     8. ext-intl
     9. ext-mbstring
     10. ext-openssl
     11. ext-pdo\_mysql
     12. ext-simplexml
     13. ext-soap
     14. ext-spl
     15. ext-xsl
     16. ext-zip
     17. lib-libxml
8. make sure PHP OPcache is enabled
9. PHP settings
   * 1. Time zone Asia/Dubai
     2. Memory limit 2GB or 4GB
     3. apache.save\_comments must enable
10. SSL
11. Required system dependencies
    * 1. Bash
      2. Gzip
      3. lsof
      4. mysql
      5. mysqldump
      6. nice
      7. php
      8. sed
      9. tar
12. Mail Server
13. Technologies Magneto can Use
    * 1. Redis version 3.2, 4.0, 5.0 for page cache and session storage
      2. Varnish version 4.x or 5.2
      3. Elasticsearch
14. Optional but recommended
    * 1. php\_xdebug 2.5.x or later for development
      2. mcrypt
      3. PHPUnit

**Magento 2 Installation**

**Step No. 2 Install Nginx for magento2**

sudo apt-get update

apt-get –y install nginx

apt-get –y install php7.2-fpm php7.2-cli

edit below both php.in file

/etc/php/7.2/fpm/php.ini

/etc/php/7.2/cli/php.in

change value of

memory\_limit 2G

max\_execution\_time = 1800

zlib.output\_compression = On

restart the php-pfm service

service restart php7.2-fpm

or systemctl restart php7.2-fpm

Open /etc/nginx/site-available/

Vim default

Make sure below lines are uncomment

location ~ \.php$ {

include snippets/fastcgi-php.conf;

fastcgi\_pass unix:/var/run/php/php7.2-fpm.sock;

**Open /etc/ngnix/site-enabled**

**Step No. 2 Install Apache2 with following command**

* 1. sudo apt-get install apache2

1. dpkg-reconfigure tzdata
   1. above command will setup date & time

2.2 Then open /etc/apache2/sites-available/000-default.conf file and add following code

<Directory “/var/www/html”>

Options Indexes FollowSymLinksMultiViews

AllowOverride All

Order allow,deny

Require all granted

</Directory>

open /etc/apache2/apache.conf and change user and Group to vagrant

open vim /etc/apache/apache.conf and make sure AllowOverride is set to All

2.3 apache2 –v

To check apache is working

2.4 sudo service apache2 status it should be in running state

create a file with name phpinfo.php in place in /var/www/html folder in Ubuntu and write following code in it

<?phpphpinfo(); ?>

and then open browser and type 127.0.0.1/phpinfo.php. should show php information

2.5 run below command so if server restart, apache2 service will

start automatically

sudo update-rc.d apache2 defaults

**Step No. 3 Install PHP version 7.1.x by running the following commands**

3.1 sudo apt-get –y update

sudo add-apt-repository ppa:ondrej/php

sudo apt-get –y update

s sudo apt-get install -y php7.1 libapache2-mod-php7.1 php7.1-common php7.1-gd php7.1-mysql php7.1-curl php7.1-intl php7.1-xsl php7.1-mbstring php7.1-zip php7.1-bcmath php7.1-iconv php7.1-soap

3.2 Run the following command to install, required PHP module

sudo apt install zip unzip php7.0-zip

3.3 install mod rewrite

sudo a2enmod rewrite

3.4 Run the following command to setup locale

locale-gen en\_US.UTF-8

export LANG=en\_US.UTF-8

export LC\_ALL=en\_US.UTF-8

3.5 To search for specific PHP module

sudo apt-cache search module\_name

sudo apt-cache search php7.1-intl

3.5 to check PHP version& installed modules in PHP

php–v

php–me

php –ini this command will show the working php.ini file

3.6 Change the following php configuration value

open php.ini file located at /etc/php/7.1/apache2/php.ini

and adjust below property

/

memory\_limit = 2BG

upload\_max\_filesize= 100M

max\_execution\_time = 360

date.timezone = Asia/Dubai

opcache.save\_comments = True

3.7 after making changes restart apache2 server

sudo service apache2 restart

**Step No. 4 Install Mysql-Server run the following commands**

4.1 run the following commands to install mysql server

sudo apt-get install -y mysql-server mysql-client

sudo service mysql start

sudo mysql\_secure\_installation

mysql–u root –p

4.2 create database in mysql with name magento2

CREATE DATABASE magento;

4.3 create user in database for magento2 with name magento2user with password m@g1t786

CREATE USER ‘magento’@’LOCALHOST’ IDENTIFIED BY ‘m@g1t786’;

GRANT ALL ON magento2.\* to ‘magento2user’@’localhost’ IDENTIFIED BY ‘m@g1t786’ WITH GRANT OPTION;

FLUSH PRIVILEGES;

4.4 run below command so if server restart, mysql service will start automatically

sudo update-rc.dmysql defaults

4.5 if root user password is not working then follow the link below to reset password

<https://www.digitalocean.com/community/tutorials/how-to-reset-your-mysql-or-mariadb-root-password-on-ubuntu-18-04>

**Step No. 5** Get your authentication key from magneto marketplace login

**Step No.6 Install GIT , CURL and COMPOSER**

sudo apt-get install curl git

sudo curl -sS https://getcomposer.org/installer | php

sudo mv composer.phar /usr/local/bin/composer

**Step No. 7 Download & Install phpMyadmin (Optional)**

Download myphpadmin from <https://www.phpmyadmin.net/downloads/>

Extract file and copy to /var/www/magento-dev.local/public

Then open browser and enter <http://localhost/phpmyadmin>

Enter username root & password m@g1t786

**Step No. 8 Setting Pre Installation Permissions**

Here’s what that means:

* 775 for directories, which means full control by the user, full control by the group, and enables everyone to traverse the directory. These permissions are typically required by shared hosting providers.
* 664 for files, which means writable by the user, writable by the group, and read-only for everyone else.

1. Create user using below commands
   * + - adduser magento\_user
       - passwd magento\_user
2. Add user to www-data group
   1. Sudo usermod -a -G www-data magento\_user
3. Check user group membership
   1. groups magento\_user
4. switch to /var/www/html directory and create a folder magento2
   1. cd /var/www/html
   2. mkdir magento2
5. change owner and grouop of magneto to magento\_user & www-data

chown –R magento\_user:www-data magento2

1. switch user from root to magento\_user and move to magento2 folder

su magento\_user

cd /var/www/html/magento2

1. Download magento software using composer using following command

composer create-project --repository=https://repo.magento.com/ magento/project-community-edition .

1. set permission on magento installation directory
   1. find . –type f –exec chmod 644 {} \; optional step
   2. find . –type d –exec chmod 755 {} \; optional step
   3. find var generated vendor pub/static pub/media app/etc-type f -execchmodg+w{} +
   4. find var generated vendor pub/static pub/media app/etc-type d -execchmodg+ws{} +
   5. sudo chown-R :www-data.
   6. chmodu+x bin/magento

Make user to change the user &Group to vagrant in /etc/apache2/apache2.conf

**Step No. 8 Installing Sample Data in Magento**

Run the belwo command from installation directory (/var/www/html)

php bin/magentosampledata:deploy

phpbin/magentosetup:upgrade

php bin/magentocache:flush

php bin/magentosetup:static-content:deploy -f

make sure RAM for VM is 3 or more GB

**Step No. 9 Run Magento Installation wizard**

now open browser and enter 192.168.20.20/magento2 and press enter

Follow the wizard to install magneto

**Step No. 10 Additional setup**

Creating a file in /etc/apache2/sites-available with name magento-dev.local.conf

Enter the following code in it

<VirtualHost \*:80>

# ServerName (domain) and admin email

ServerAdminwebmaster@magento-dev.local

ServerNamemagento-dev.local

DocumentRoot /var/www/magento-dev.local/public

# above folder of the site, we have to create this folder

# log file locations

LogLevelwarn

ErrorLog /var/log/apache2/magento-dev.error.log

CustomLog /var/log/apache2/magento-dev.access.log combined

</VirtualHost>

enable the new virutalhost

from sites-availble folder location, enter following command to enable the site

sudo a2ensite magento-dev.local.conf

and to disable the default

sudo a2dissite 000-default.conf

restart the apache2 service and test the site.

**Step No. 11 Cloning from Github**

1. initialize git repository in /var/www/html/magento2

git init

1. issue below command to ignore the existing changes

$ git checkout master  
$ git fetch --all  
$ git reset --hard origin/master

Creating a file in /etc/apache2/sites-available with name magento-dev.

**Store Setup &Configurations**

**Entering Company Information**

Login to admin 🡪 click Stores🡪 click Configuration 🡪 click General tab 🡪 click general 🡪

country option 🡪 select default United Arab Emirates 🡪 select allow United Arab Emirates 🡪 Locale Options 🡪 select timezone (Gulf standard Time (Asia/Dubai) 🡪 Select Locale English United States 🡪 Weight Unit (Kgs) 🡪Store Information (Enter the name & other details )

click General tab 🡪 click web 🡪 click Base Urls& Base Urls (secure)🡪 Enter the site url🡪

make sure putting / at end of url.

click General tab 🡪 click Currency 🡪 select Base , Default and Allowed Current to United Arab

Emirates Dirham.

click General tab 🡪 click Store Email Addresses 🡪 Enter in Sender Name (Eye Glassess Online

Customer Care ) 🡪 Enter Email address [sales@eyeglassesonline.com](mailto:sales@eyeglassesonline.com)🡪

click General tab 🡪 click contacts 🡪 Enter Email Address and Other info

Leave the other tab in General as it as.

Login to admin 🡪 click Stores🡪 click Configuration 🡪 click Catalog tab 🡪 Click Catalog 🡪 click store front 🡪 use flat catalog product (yes) 🡪 Change Product alert for product comes back in store (yes) 🡪 Search Engine Optimization 🡪 use Canonical link meta tag for products (yes) 🡪

Click catalog 🡪 inventory 🡪 display out of stock product (yes ) 🡪 set only x left threshold (2) 🡪

Login to admin 🡪 click Stores🡪 click Configuration 🡪 click Sales tab 🡪 Click Sales 🡪 Invoice and Packing Slip Desgin🡪 Upload Logo 🡪 Enter Address 🡪

**Entering Terms & Conditions**

Login to admin 🡪 click Stores🡪 click terms & conditions 🡪 click add terms & condition 🡪 give name (agreements) 🡪 Status Enable 🡪 Show content as (Text ) 🡪 applied (Manually) 🡪 Store View (All Stores ) 🡪Checkboxk Text (I Agreed) 🡪 Enter the Term & Condition Body

**Setup Tax Rules**

Login to admin 🡪 click Stores🡪click tax zones & rates🡪click add tax rules 🡪 give name UK 🡪 zip code (\*) 🡪 Counter (United Arab Emirates) 🡪 Rate % (5) 🡪 click save 🡪repeate the process for Dubai

Login to admin 🡪 click Stores🡪 click Tax Rules 🡪 click add 🡪 give name 🡪 tax rates select UK 🡪 Save

Creating XML Site Map

Login to admin 🡪 click Stores🡪 click Configuration 🡪 click Catalog tab 🡪 XML site map --

Click generation setting (yes ) 🡪 now click marketing 🡪 click site map 🡪 click add 🡪 give name (sitemap.xml) 🡪 path enter (/sitemap/) 🡪before saving make sure a directory with name sitemap is created at root of webserver with permission 770 and group (www-data) 🡪save

**Changing Logo and Footer**

Logo Changing

Login to admin 🡪 click content 🡪 click Configuration 🡪 click edit 🡪 click header 🡪 upload logo

Footer Changing

Login to admin 🡪 click content 🡪 click Configuration 🡪 click edit 🡪 click Footer 🡪 change it accordingly.

**Changing home page layout**

Login to admin 🡪 click content 🡪 click pages 🡪 Select Home Pages🡪 click edit 🡪 change it to one column in design tab

**Creating Category & Sub Category**

Login to admin 🡪 click Catalog 🡪 click Categories 🡪 Select default category

🡪 Click add sub Category🡪 finish all Category and sub category 🡪 this also work as menu bar for site.

**Add / Creating Sample Products**

Login to admin 🡪 click Catalog 🡪 click Products 🡪 click add product 🡪 from

drop down select simple product 🡪Give Product name , Price , SKU, Images etc

**How to create static banner on home page**

Login to admin 🡪 click content 🡪 click Block 🡪 create new 🡪 Static Bannar

name 🡪 click insert image 🡪 save block

To Display block on home page 🡪 we need to create widgets 🡪 Login to admin 🡪

click content 🡪 click widgets 🡪 select static block 🡪 select the location as

home page. 🡪 if banner is not display clear the cache

if we don’t want this block then delete the block and the widget both.

**How to create Slidder on Home Page**

Download Slidder Extension from magento marketplace 🡪 login to marketplace and

search for banner 🡪 make sure it’s compatible with your magento version 🡪 add to cart and check out 🡪 click install 🡪 this will give you access key 🡪 go to admin in magento🡪 click System 🡪 click web setup wizard 🡪 click extension manager 🡪enter the access key 🡪 follow the wizard

Configure the extension from admin

Download bestseller extension and configure to display bestselling item on

home page.

How to create static Promotional banner block on home page

After cloning the repository

phpbin/magentosetup:db-schema:upgrade

phpbin/magentosetup:db-data:upgrade

phpbin/magentocache:clean

php bin/magentoadmin:user:create

php bin/magentosetup:static-content:deploy –f

git reset --hard to discard the local change on production server

git pull to pull the latest modification

Installing Captranio

sudo apt-get update

sudo apt-get install build-essential

sudo apt-get install curl

gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3 7D2BAF1CF37B13E2069D6956105BD0E739499BDB

\curl -sSL [https://get.rvm.io](https://get.rvm.io/) | bash -s stable

source ~/.rvm/scripts/rvm

rvm install 2.5.3

rvm use 2.5.3

rvm use 2.5.3 –default

ruby -v

gem install capistrano-magento2

Go to project direction 🡪 create a file with name Gemfile using command

Sudo touch Gemfile and add the following code

source 'https://rubygems.org'

gem 'capistrano-magento2'

Run below command to install the gem.

bundle install

now in project directory create run following commands

mkdir -p tools/cap

This command will create the following folder structure

Capfile

config

deploy

production.rb

staging.rb

deploy.rb

lib

caistrano

taskl

open Capfile and enter following details

# Load DSL and set up stages

require 'capistrano/setup'

# Load Magento deployment tasks

require 'capistrano/magento2/deploy'

require 'capistrano/magento2/pending'

# Load Git plugin

require "capistrano/scm/git"

install\_plugin Capistrano::SCM::Git

# Load custom tasks from `lib/capistrano/tasks` if you have any defined

Dir.glob('lib/capistrano/tasks/\*.rake').each { |r| import r }

Open config/deploy.rb file and enter following details

set :application, "IYElectronics\_store"

set :repo\_url, [git@github.com:rizwanwakil2018/magento-production.git](mailto:git@github.com:rizwanwakil2018/magento-production.git)

Open config/deploy/production.rb file and enter following details

server "35.187.71.113", user: "www-data", roles: %w{app db web} #, my\_property: :my\_value

# server "example.com", user: "deploy", roles: %w{app web}, other\_property: :other\_value

# server "db.example.com", user: "deploy", roles: %w{db}

set :deploy\_to, '/var/www/html'

set :branch, proc { `git rev-parse --abbrev-ref master`.chomp }

set :magento\_deploy\_languages, ['en\_US', 'en\_CA']

set :magento\_deploy\_composer, false

make a copy of auth.json.sample to auth.json and add the github access key, magneto access key & password and save the file.

Magneto key & password can be found in login to marketplace and in profile

Github access can by found in github setting 🡪 developer setting 🡪 personal access key

##### **SSH**

Capistrano deploys using SSH. Thus, you must be able to SSH (ideally with keys and ssh-agent) from the deployment system to the destination system for Capistrano to work.

Issue the following command to check the github connectivity with local server

ssh -T git@github.com

generate the ssh key using following command

ssh-keygen -f ~/.ssh/dev\_ssh\_key -t rsa -b 4096 -C "vagrant"

now add the ssh key to github repository

open the id\_rsa.pub file and copy the content and paste in github 🡪 open repo 🡪 settings 🡪 deploy keys 🡪 give name and paste over there

now to check below command

ssh -T git@github.com

now rename the .gitignore file to some thing

create a repository in github

push all the content from local server to github repository.

now install the google cloud compute on local system to copy the id\_rsa.pub file to remove server.

# Import the Google Cloud Platform public key  
curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

# Update the package list and install the Cloud SDK  
sudo apt-get update && sudo apt-get install google-cloud-sdk

gcloud init

now try to connect to remote server

gcloud compute ssh magento2

now copy id\_rsa.pub file from local system to remote server

gcloud compute scp ~/ssh/id\_rsa.pub magento2:~

gcloud compute scp [LOCAL\_FILE\_PATH] [INSTANCE\_NAME]:~ command format

gcloud compute scp vagrant@magento2:~/.ssh/id\_rsa .

now copy the content of id\_rsa.pub file and open instance page goole cloud admin and add SSH Key and add the content.

To check connect issue below command

ssh vagrant@ [35.187.71.113](https://35.187.71.113/)

Step No. 1 Create a Folder in Mac System with magento\_dev\_env

Step No. 2 Paste a Vagrantfile in this follower (Vagrantfile available on github repo

rizwanwakil2018/studynotes

Step No. 3 run the vagrant up command to create VM based on Vagrantfile

Step No. 4 configure apache2, php , mysql in VM using above step in this documents

Step No. 5 create a new folder with name development inside magento\_dev\_env folder

Step No. 6 download magento software in this development folder using composer on MAC system

Map this development folder with VM machine /var/www/html/magento2/ using

Vagrantfile

Step No. 7 Install the magento site locally

Step No. 8 create repository with name magento-store in github

Step No. 9 push all the development content to repository

Step No. 10 take magento local database backup and restore on production server

<https://www.linode.com/docs/databases/mysql/use-mysqldump-to-back-up-mysql-or-mariadb/>

mysqldump -u root –p magento > dev\_db\_backup.sql