```
# Puppet Site Manifest: wordPressInstall.pp
#
# Manifest installation instructions:
# 1. Copy this file to directory: /etc/puppetlabs/code/environments/production/manif
ests/
# 2. Installs complete WordPress installation to specified nodes.
# 3. Run:
#      puppet apply /etc/puppetlabs/code/environments/production/manifests/wordPress
Install.pp
#
#########################################################################
## PREINSTALLATION STEPS AND CONSIDERATIONS BEFORE SETTING UP THE NODE ##
#########################################################################
#
# 1. Consider using puppetServerInstall.sh
#    The puppetServerInstall.sh script creates the files listed in section 3. PREREQ
 SERVER FILES:
#    https://github.com/linuxfjb/PuppetWordPressConfigMgmtProject/blob/main/puppetSe
rverInstall.sh
#
# 2. Consider editing or changing the default MySQL password in this script.
#    VERY IMPORTANT: It is recommended to change this after installation.
#
# 3. PREREQ SERVER FILES - FMI check out: https://github.com/linuxfjb/PuppetWordPres
sConfigMgmtProject
#    1. Apache server set up: wordpress.conf
#      server:            /home/ubuntu/wordpress_configfiles/wordpress.conf
#      node destination:  /etc/apache2/sites-available/wordpress.conf
#
#    2. WordPress config vars and settings: wp-config.php
#      server:            /home/ubuntu/wordpress_configfiles/wp-config.php
#      node destination:  /srv/www/wp-config.php
#
#    3. WordPress db creation and db user setup script: wordpressMySQL.sql
#      server:           /home/ubuntu/wordpress_configfiles/wordpressMySQL.sql
#      node:  MySQL DB: wordpress
#
#    4. Edit the wp-config.php file.
#       Go to https://api.wordpress.org/secret-key/1.1/salt/ and copy encrypted vari
ables
#       into the authentication variables section AUTH_KEY, SECURE_AUTH_KEY, NONCE_K
EY, etc...
#
#    5. After installation:
#       a. Consider changing the default MySQL password on the node.
#       b. Consider changing the password to the wordpress DB User.
#       c. Consider removing the /root/wordpressMySQL.sql installation file or at le
ast scrub
#          the password. This could be automated in this manifest.
#
###############################
### MYSQL DEFAULT PASSWORD ###
###############################
class base::wputil_configvars {
  $mysql_password = 'mypass'
}

class base::apache_running {
  package { 'apache2' :
    ensure => 'present'
  }

  service { 'apache2' :
    ensure => 'running',
    enable => 'true'
  }
}

class base::wputil_webpackage {

  package {'libapache2-mod-php' :
    ensure => 'present'
  }
```

```
  package {'php' :
    ensure => 'present'
  }

  package {'php-bcmath' :
    ensure => 'present'
  }

  package {'php-curl' :
    ensure => 'present'
  }

  package {'php-imagick' :
    ensure => 'present'
  }

  package {'php-intl' :
    ensure => 'present'
  }

  package {'php-json' :
    ensure => 'present'
  }

  package {'php-mbstring' :
    ensure => 'present'
  }

  package {'php-mysql' :
    ensure => 'present'
  }

  package {'php-xml' :
    ensure => 'present'
  }

  package {'php-zip' :
    ensure => 'present'
  }
}

class base::wputil_ghostscriptpackage {
  package { 'ghostscript' :
    ensure => 'present'
  }
}

class base::wputil_mysqlpackage {
  package {'mysql-server' :
    ensure => 'present'
  }
}

# Download the wordpress installation into /tmp directory than copy over
# into /srv/www per web site instructions.
# You may want to clean up /tmp directory.
class base::wputil_download_wp {

  # Make sure destination directory for wordpress is created.
  file { '/srv/www' :
    ensure => 'directory',
    owner => 'www-data',
    group => 'www-data'
  }

  # BEGIN
  # curl https://wordpress.org/latest.tar.gz | sudo -u www-data tar zx -C /srv/www
  # BEGIN

  # Download the Wordpress bundle from puppet module server.
  # Note: the File puppet command was having a checksum issue downloading from:
  # puppet:///modules/wordpress/latest.tar.gz so just used wget instead.
```

```
  exec { 'wget':
    cwd => "/tmp",
    command => "wget https://wordpress.org/latest.tar.gz",
    path => ['/bin'],
  }~>
  # Ensure the file is downloaded.
  file { '/tmp/latest.tar.gz':
    ensure => file,
    owner => 'www-data',
    group => 'www-data',
  }~>
  # Untar wordpress install file.
  exec { 'extract':
    cwd => "/tmp",
    command => "tar -xvzpf latest.tar.gz",
    path => ['/bin'],
    require => File['/tmp/latest.tar.gz'],
  }

  # Bring in the wp-config.php file into the tmp/wordpress directory to be copied.
  file { '/tmp/wordpress/wp-config.php':
    content => template('/home/ubuntu/wordpress_configfiles/wp-config.php'),
  }

  # This command copies to /srv/www
  # after the wordpress directory was untarred inside the /tmp directory.
  exec { 'cp -r /tmp/wordpress /srv/www/':
    path => ['/bin'],
    require => Exec['extract']
  }~>
  exec { 'chown -R www-data:www-data /srv/www/wordpress':
    path => ['/bin'],
    require => Exec['cp -r /tmp/wordpress /srv/www/'],
  }

  # Directory should be there with the correct ownership.
  file { '/srv/www/wordpress':
    ensure => 'directory',
    owner => 'www-data',
    group => 'www-data',
    require => Exec['chown -R www-data:www-data /srv/www/wordpress'],
  }

  # END
  # curl https://wordpress.org/latest.tar.gz | sudo -u www-data tar zx -C /srv/www
  # END
}

# Enable wordpress site, disable default "It Works" site.
# A call will be made to 'apache2' service from base::wputil_apachestart.
# If you want, verify site enabled: apache2ctl -S
class base::wputil_enable_apacheweb {
  #Note that changing wordpress.conf will trigger this section including
  #resetting mysql password.
  file { '/etc/apache2/sites-available/wordpress.conf':
    content => template('/home/ubuntu/wordpress_configfiles/wordpress.conf'),
  }~>
  exec { 'enable vhost file':
    command => '/usr/sbin/a2ensite wordpress.conf',
    refreshonly => true,
    #path => ['/usr/sbin'],
    require => File['/etc/apache2/sites-available/wordpress.conf'],
  }~>
  exec { 'disable "It Works" site':
    command => '/usr/sbin/a2dissite 000-default',
    refreshonly => true,
    notify => Service['apache2'],
    require => File['/etc/apache2/sites-available/wordpress.conf'],
  }~>
  #Changes mysql password.
  #Note: This command is intentionally placed here as to not run this after
  #wordpress site is already set up in browser!
  exec { 'set mysql root password':
```

```
    command => 'mysqladmin -u root password $base::wputil_configvars::mysql_password
',
    path => ['/bin'],
    require => Package['mysql-server'],
  }

}

# MySQL setup for wordpress site.
# Requires package mysql-server is installed from base::wputil_mysqlpackage.
# Warning: Clear text password is used. Scrub when done. It's on you if you don't.
#          Also, clean up root directory. It's on you if you don't.
class base::wputil_mysqlconfiguration {

  file { '/root/wordpressMySQL.sql':
    content => template('/home/ubuntu/wordpress_configfiles/wordpressMySQL.sql'),
  }~>
  #execute database script
  exec { 'mysql database':
    command => 'mysql -u root -p$base::wputil_configvars::mysql_password -h localhos
t < /root/wordpressMySQL.sql',
    path => ['/bin'],
    #Note:
    #Returns [1] quiets the output - In the event this exec is run more than once du
e to interval,
    #create database sql will create error.
    #Thus, the sql will back out and not overwrite any changes in the DB.
    #Not elegant but proven not to interfere with site if run more than once.
    returns => [0,1],
  }

  # Finally, enable mysql for the wordpress site to start working
  service {'mysql' :
    ensure => 'running',
    enable => 'true'
  }
}

class base::wputil_install_mysql {
  include base::apache_running, base::wputil_webpackage, base::wputil_ghostscriptpac
kage, base::wputil_mysqlpackage
  include base::wputil_enable_apacheweb, base::wputil_download_wp, base::wputil_mysq
lconfiguration
}

################# NODES #################

node  'default' {
}

# Edit the nodes that need WordPress installation here.
node 'client1.ec2.internal' {
  include base::wputil_install_mysql
}
```