

# Backup and Recovery

To create the backup of the website "Techvish", you have to take a copy of mainly

- DATABASE
- Html files (wp-config.php)

Rest of the server backup can be taken if needed, in case you have any other important files saved in the server. Here I only needed my site backup, so I took only the html files and the database backup.

## Backup :

Created a volume "Backup" of size 5GB in the same availability zone (1-a)

Attached it to the instance where my website is hosted .

Named it as /dev/xvdd for mounting

Formatted the volume – "`#mkfs.ext4 /dev/xvdd`"

Created a mountpoint – "`#mkdir /mnt/mybackup`"

Mounted the volume – "`#mount /dev/xvdd /mnt/mybackup`"

Created a backup of html directory using gzip –

"`#tar -czf backupofhtml.gz /var/www/html`"

Created a backup of the database of my website, here named "wordpress",  
username – vishnu & password "free....." –

"`#mysql -u root -p`"

"`mysqldump -u your_username -p your_database_name > /path/to/backup/directory/db_backup.sql`"

Now send both the files to your volume using cp command

**"cp /pathtobackupfiles /mnt/mybackup"**

Now list is to make sure , you got the proper files and then unmount the volume

**#umount /dev/xvdd**

## Recovery :

Step 1) Create an Instance and repeat the installation of LAMP stack and do other necessary updates and installations if needed

### Install LAMP Stack (Linux, Apache, MySQL, PHP)

**# yum install httpd php php-mysqld -y**

```
root@ip-172-31-12-160 ~# yum install httpd php php-mysqld -y
Last metadata expiration check: 0:03:03 ago on Wed Oct 18 09:14:33 2023.
Dependencies resolved.
=====================================================================================================================================
Package                               Architecture      Version            Repository         Size
=====================================================================================================================================
Installing:
httpd                                 x86_64            2.4.56-1.amzn2023 amazonlinux        48 k
php8.2                               x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        13 k
php8.2-mysqld                        x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        150 k
Installing dependencies:
apr                                   x86_64            1.7.2-2.amzn2023.0.2 amazonlinux        129 k
apr-util                             x86_64            1.6.3-1.amzn2023.0.1 amazonlinux        98 k
generic-logos-httpd                 noarch            18.0.0-12.amzn2023.0.3 amazonlinux        19 k
httpd-core                           x86_64            2.4.56-1.amzn2023 amazonlinux        1.4 M
httpd-filesystem                     noarch            2.4.56-1.amzn2023 amazonlinux        15 k
httpd-tools                          x86_64            2.4.56-1.amzn2023 amazonlinux        82 k
libbrotli                             x86_64            1.0.9-4.amzn2023.0.2 amazonlinux        315 k
libbrotli-unicode                    x86_64            1.0.18-13.amzn2023.0.1 amazonlinux        166 k
libbrotli-unicode-icu                 x86_64            1.1.34-1.amzn2023.0.2 amazonlinux        241 k
libldap                              noarch            2.1.49-3.amzn2023.0.3 amazonlinux        33 k
nginx-filesystem                     noarch            1.11.24.0-1.amzn2023.0.2 amazonlinux        9.1 k
php8.2-cli                           x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        3.6 M
php8.2-common                        x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        692 k
php8.2-gd                             x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        90 k
php8.2-ldap                           x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        45 k
php8.2-xml                            x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        149 k
Installing weak dependencies:
apr-util-openssl                     x86_64            1.6.3-1.amzn2023.0.1 amazonlinux        17 k
mod_http2                            x86_64            2.0.11-2.amzn2023 amazonlinux        150 k
mod_lua                              x86_64            2.4.56-1.amzn2023 amazonlinux        62 k
php8.2-fpm                           x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        1.9 M
php8.2-mcrypt                         x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        126 k
php8.2-opcache                       x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        360 k
php8.2-sodium                         x86_64            8.2.9-1.amzn2023.0.3 amazonlinux        43 k
Transaction Summary
-----
Install 26 Packages
Total download size: 10 M
Installed size: 44 M
Downloading Packages:
(1/25): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm                                1.3 MB/s | 98 kB  00:00
```

### Start and enable the services

**#systemctl start httpd**

**#systemctl enable httpd**

```
[root@ip-172-31-12-160 html]# systemctl start httpd
[root@ip-172-31-12-160 html]# systemctl enable httpd
```

To install MariaDB 10.5 server on your Amazon Linux instance using yum, you can follow these steps:

1. Connect to your EC2 instance: Use SSH to connect to your Amazon Linux instance.
2. Update your package manager: Before installing any new packages, it's a good idea to update your package manager to ensure you're getting the latest versions of the software. Run the following command:

```
'''
```

```
#sudo yum update
```

```
'''
```

3. Install MariaDB 10.5 server: Run the following command to install MariaDB 10.5 server:

```
'''
```

```
#sudo yum install mariadb105-server
```

```
'''
```

4. Start the MariaDB service: After the installation is complete, start the MariaDB service using the following command:

```
'''
```

```
#sudo systemctl start mariadb
```

```
'''
```

5. Enable MariaDB to start on boot: If you want MariaDB to start automatically whenever the system boots, run the following command:

'''

```
#sudo systemctl enable mariadb
```

'''

6. Secure your MariaDB installation: It's recommended to run the MySQL/MariaDB security script to secure your installation. Run the following command and follow the prompts:

'''

```
#sudo mysql_secure_installation
```

'''

7. Verify the installation: You can verify that MariaDB is running by checking its status:

'''

```
#sudo systemctl status mariadb
```

'''

That's it! You've now installed MariaDB 10.5 server on your Amazon Linux instance. You can start using it as your database server.

Installation of wordpress

```
# yum install wget -y
```

```
[root@rhel-01-01-181 ~]# ./cpr [var/www/html/  
[root@rhel-01-01-181 ~]# curl -m 10 --insecure http://  
Warning: certificate management repository:  
Unable to read consumer identity  
  
This system is not registered with an entitlement server. You can use subscription-manager to register.  
  
Last metadata expiration check: 6:10:29 ago on Tue 17 Oct 2023 01:40:10 PM UTC.  
Dependencies resolved.  
Package Architecture Version Repository Size  
--  
Installing:  
cppt  x86_64 1.21.1-7.el9 rhel-9-appstream-rhel-cym 794 k  
Transaction Summary  
Install: 1 Package  
Total download size: 794 k  
Installed size: 8.1 M  
Isosetting packages:  
cppt-1.21.1-7.el9.x86_64.rpm 16 MB/s | 794 kB | 00:00  
Total 9.4 MB/s | 794 kB | 00:00  
Running transaction check  
Transaction check succeeded.  
Running transaction test  
Transaction test succeeded.  
Running transaction  
Preparing :  
Installing : cppt-1.21.1-7.el9.x86_64 1/1  
Running scriptlet: cppt-1.21.1-7.el9.x86_64 2/1  
Verifying : cppt-1.21.1-7.el9.x86_64 3/1  
Installed products updated.  
Installed:  
cppt-1.21.1-7.el9.x86_64  
Complete!  
[root@rhel-01-01-181 ~]#
```

```
# wget https://wordpress.org/latest.zip
```

```
#unzip latest.zip
```

[illegible]

Now copy all the contents of wordpress directory to default document root (/var/www/html)

```
# cp -r wordpress/* /var/www/html/
```

```
#ls
```

```
[root@ip-172-31-31-131 html]# cp -r wordpress/* /var/www/html/
[root@ip-172-31-31-131 html]# ls
index.php      readme.html    wp-admin      wp-blog-header.php  wp-config.php  wp-cron.php  wp-load.php  wp-settings.php  xmlrpc.php
latest.tar.gz  wordpress     wp-activate.php  wp-comments-post.php  wp-content     wp-links-opml.php  wp-mail.php  wp-trackback.php
license.txt    wp-activate.php  wp-comments-post.php  wp-content
```

Also remove other files

```
#rm -rf latest.zip
```

```
[root@ip-172-31-12-160 html]# rm -rf latest.zip
[root@ip-172-31-12-160 html]# ls
wordpress
```

```
#rm -rf wordpress
```

```
#ls
```

```
wp-activate.php  wp-content  wp-login.php  xmlrpc.php
[root@ip-172-31-12-160 html]# rm -rf wordpress/
[root@ip-172-31-12-160 html]# ls
index.php      wp-blog-header.php  wp-includes  wp-settings.php
license.txt    wp-comments-post.php  wp-links-opml.php  wp-signup.php
readme.html    wp-config-sample.php  wp-load.php  wp-trackback.php
wp-activate.php  wp-content  wp-login.php  xmlrpc.php
wp-admin        wp-cron.php  wp-mail.php
```

Set ownership as follows

```
#chmod -R 755 wp-content
```

```
#chown -R apache:apache wp-content
```

```
wp-config-sample.php  wp-cron.php
[root@ip-172-31-12-160 html]# chmod -R 755 wp-content
[root@ip-172-31-12-160 html]# chown -R apache:apache wp-content
[root@ip-172-31-12-160 html]#
```

Restart http service

```
#systemctl restart httpd
```

```
[root@ip-172-31-35-131 html]#
[root@ip-172-31-35-131 html]#
[root@ip-172-31-35-131 html]# systemctl restart httpd
[root@ip-172-31-35-131 html]#
[root@ip-172-31-35-131 html]#
[root@ip-172-31-35-131 html]#
```

Now the pre-requisites are done ,

Attach the volume Backup where you have stored the files to the instance ,

```
#mount /dev/xvdd /mnt/mybackup
```

Replace the html files with the html files in the backup ,

**#tar -xzf backupofhtml.gz**

Replace the Html files with these new files

(you can either use tar for backup or just cp if the files aren't too big ,so if you used cp command , you can directly copy paste instead of extracting)

**WordPress Files:** These are the files that make up your WordPress website. They include core WordPress files, themes, plugins, and uploads. The main directory you'll want to back up is usually located in the web server's document root directory. In a typical LAMP (Linux, Apache, MySQL, PHP) setup, this directory is often **/var/www/html** or **/var/www**.

You can create a backup of the WordPress files by copying the entire WordPress directory to a safe location. For example:

```
bash
```

```
cp -r /var/www/html /path/to/backup/directory
```

## Now for the Database Recovery ,

1. **Transfer the Backup File:** If your backup file (**db\_backup.sql**) is stored on a different server or location, transfer it to the server where your WordPress website is hosted. You can use tools like SCP, SFTP, or FTP for this purpose.
2. **Access the MySQL Command Line:** Log in to your server via SSH and access the MySQL command line interface. You can do this by running the following command and entering your MySQL root password when prompted:

**#mysql -u root -p**

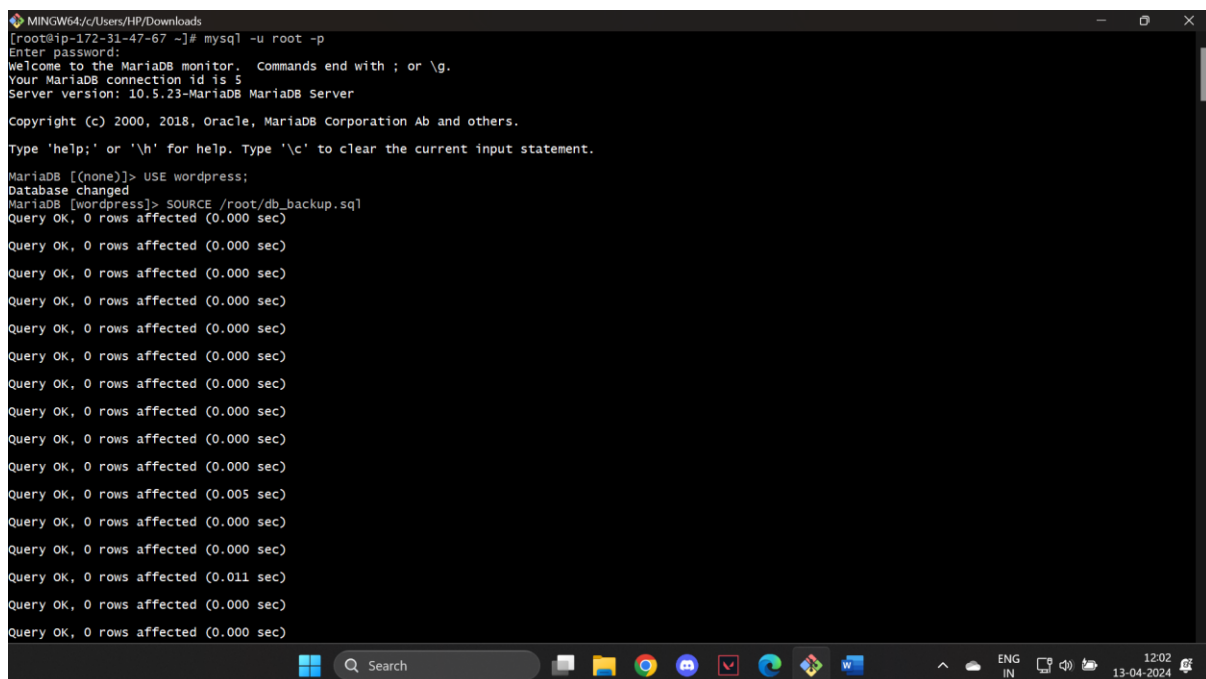
**Create a New Database (Optional):** If you want to restore the backup to a new database, you can create one using the following MySQL command:

**#CREATE DATABASE new\_database\_name;**

**Restore the Database:** Once you're in the MySQL command line interface and have selected the appropriate database (if necessary), you can restore the database from the backup file using the following command:

**#USE your\_database\_name;**

**#SOURCE /path/to/backup/directory/db\_backup.sql;**

A screenshot of a Windows terminal window titled 'MINGW64/c/Users/HP/Downloads'. The terminal shows a MySQL command-line session. The user runs 'mysql -u root -p' and enters a password. The prompt changes to 'MariaDB [(none)]>'. The user runs 'USE wordpress;' and the prompt changes to 'MariaDB [wordpress]>'. The user then runs 'SOURCE /root/db\_backup.sql'. The terminal shows a series of 'Query OK, 0 rows affected (0.000 sec)' messages, indicating the successful restoration of the database. The Windows taskbar is visible at the bottom, showing the time as 12:02 on 13-04-2024.

```
MINGW64/c/Users/HP/Downloads
[root@ip-172-31-47-67 ~]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 5
Server version: 10.5.23-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> USE wordpress;
Database changed
MariaDB [wordpress]> SOURCE /root/db_backup.sql
Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.005 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.011 sec)

Query OK, 0 rows affected (0.000 sec)

Query OK, 0 rows affected (0.000 sec)
```

## Check Database Credentials:

- Open your **wp-config.php** file located in your WordPress root directory.
- Verify that the database name, database username, and database password are correct. These details should match the credentials for the database you restored.



php

```
define('DB_NAME', 'your_database_name');  
define('DB_USER', 'your_database_username');  
define('DB_PASSWORD', 'your_database_password');
```

### Verify Database Host:

- Ensure that the database host is correctly set. For most cases, it's **localhost**, but if your database is hosted on a different server, you'll need to specify its address

php

```
define('DB_HOST', 'localhost');
```

### Database Permissions:

- Make sure that the database user has the necessary permissions to access and modify the database. You might need to grant permissions to the database user if they were not restored along with the database.

```
#GRANT ALL PRIVILEGES ON your_database_name.* TO  
'your_database_username'@'localhost' IDENTIFIED BY  
'your_database_password';
```

```
#FLUSH PRIVILEGES;
```

Now restart httpd

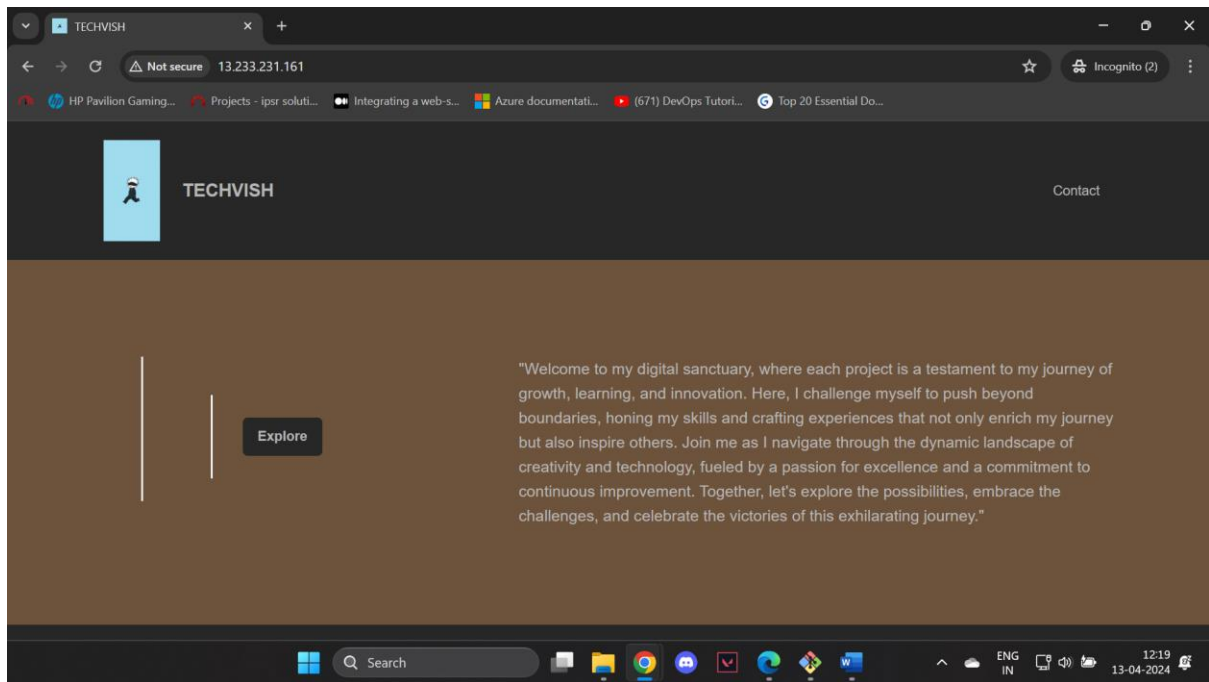
## #systemctl restart httpd

That's it! Your WordPress database should now be restored from the backup file. You can now proceed to configure your WordPress site to use this database if necessary.

Instances (2) <a href="#">Info</a>							
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>				All states ▾		< 1 > ⚙	
<input type="checkbox"/>	Name <a href="#">✎</a>	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm stat	
<input type="checkbox"/>	Backup Test	i-0b592bf846c49cc34	✔ Running 🔍	t2.micro	✔ 2/2 checks p:	<a href="#">View alarm</a>	
<input type="checkbox"/>	Project Amazon TECHVISH	i-040add33ff401a489	✔ Running 🔍	t2.micro	✔ 2/2 checks p:	<a href="#">View alarm</a>	

<a href="#">EC2</a> > <a href="#">Instances</a> > i-0b592bf846c49cc34		
<b>Instance summary for i-0b592bf846c49cc34 (Backup Test)</b> <a href="#">Info</a>		
<input type="button" value="Refresh"/> <input type="button" value="Connect"/> <input type="button" value="Instance state ▾"/> <input type="button" value="Actions ▾"/>		
Updated less than a minute ago		
Instance ID 📄 i-0b592bf846c49cc34 (Backup Test)	Public IPv4 address 📄 13.233.231.161 <a href="#">open address</a> 🔗	Private IPv4 addresses 📄 172.31.47.67
IPv6 address -	Instance state ✔ Running	Public IPv4 DNS 📄 ec2-13-233-231-161.ap-south-1.compute.amazonaws.com <a href="#">open address</a> 🔗
Hostname type IP name: ip-172-31-47-67.ap-south-1.compute.internal	Private IP DNS name (IPv4 only) 📄 ip-172-31-47-67.ap-south-1.compute.internal	Elastic IP addresses -
Answer private resource DNS name -	Instance type t2.micro	

Access the ip address



This is the backup of my website !

Now I can use the backup files in the volume to recover my website incase the instance gets corrupted or destroyed.

another safe practice is to write a backup script which runs daily or weekly and stores the data in cloud or s3 bucket ,s o that you can have access to the latest data instead of having to do all these steps manually.