# Multi-Service Server Deployment and Maintenance with Automation

Objective:
Create a Linux server environment that:

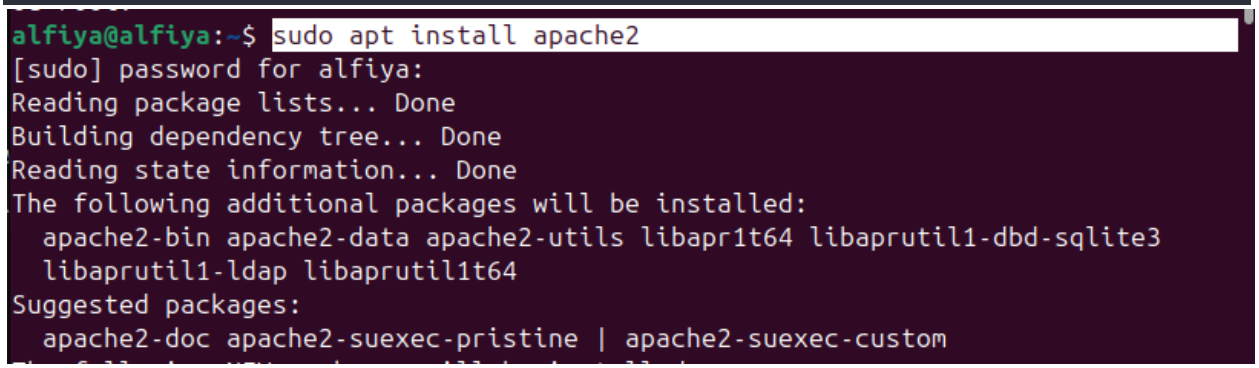
1. Hosts a website using Nginx and Apache2.
2. Manages users with specific roles and permissions.
3. Uses a database MySQL for dynamic website content.
4. Automates backup of website data, configuration files, and databases further compressing and archiving it using shell scripts.
5. Automating backups using cron jobs.

1. Web Hosting with Apache and Nginx

Objective: Set up Apache as the primary web server and Nginx as a reverse proxy.
Steps:

1. Install Apache and Nginx:

```
sudo apt install apache2 nginx
```




2. Configure Apache to use port 8080:

- Edit the Apache configuration:

```
sudo nano /etc/apache2/ports.conf
```


- Change Listen 80 to:

```
  GNU nano 7.2                                                    /etc/apache2/p
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
        Listen 443
</IfModule>
```

- Listen 8080

```
  GNU nano 7.2                                                    /etc/apache2/p
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8080

<IfModule ssl_module>
        Listen 443
</IfModule>
```

- Edit the default virtual host:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

- Change <VirtualHost *:80> to:

```
  GNU nano 7.2                                          /etc/apache2/sites-availa
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port tha
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com
```

- <VirtualHost *:8080>

```
 GNU nano 7.2                                                /etc/apache2/sites-availa
<VirtualHost *:8080>
        # The ServerName directive sets the request scheme, hostname and port th
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com
```

- Restart Apache:

```
sudo systemctl restart apache2
```

```
alfiya@alfiya:~$ sudo systemctl restart apache2

alfiya@alfiya:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabl
     Active: active (running) since Sat 2024-11-30 19:36:13 CST; 12s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 8719 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS
   Main PID: 8723 (apache2)
      Tasks: 55 (limit: 38038)
     Memory: 5.6M (peak: 7.5M)
        CPU: 39ms
     CGroup: /system.slice/apache2.service
```

3. Configure Nginx as a reverse proxy:

- Edit the Nginx configuration:

```
sudo nano /etc/nginx/sites-available/default
```

- Update the server block:

```
server {
    listen 80;
    server_name yourdomain.com;

    location / {
        proxy_pass http://127.0.0.1:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
```

```
}
```

```
#       }
server {
    listen 80;
    server_name testnginx.com www.testnginx.com;

    location / {
        proxy_pass http://127.0.0.1:8080; #forwards to apache
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}



        # pass PHP scripts to FastCGI server
```

- Test and restart Nginx:

```
sudo nginx -t
sudo systemctl restart nginx
```

```
alfiya@alfiya:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
alfiya@alfiya:~$
```

```
alfiya@alfiya:~$ sudo systemctl restart nginx
alfiya@alfiya:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-11-30 19:51:05 CST; 7s ago
       Docs: man:nginx(8)
    Process: 9180 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 9181 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 9183 (nginx)
      Tasks: 13 (limit: 38038)
```

4. Update /etc/hosts File for Local Testing

- testnginx.com add it to your local /etc/hosts file.
- Edit the file:

```
sudo nano /etc/hosts
```

```
Nov 30  20:25
alfiya@alfiya: ~
  GNU nano 7.2                                  /etc/hosts
127.0.0.1 localhost
127.0.1.1 alfiya
127.0.0.1 testnginx.com

# The following lines are desirable for IPv6 capable hosts
```

5. Verify which ports are in use:

- Use ss to check ports:

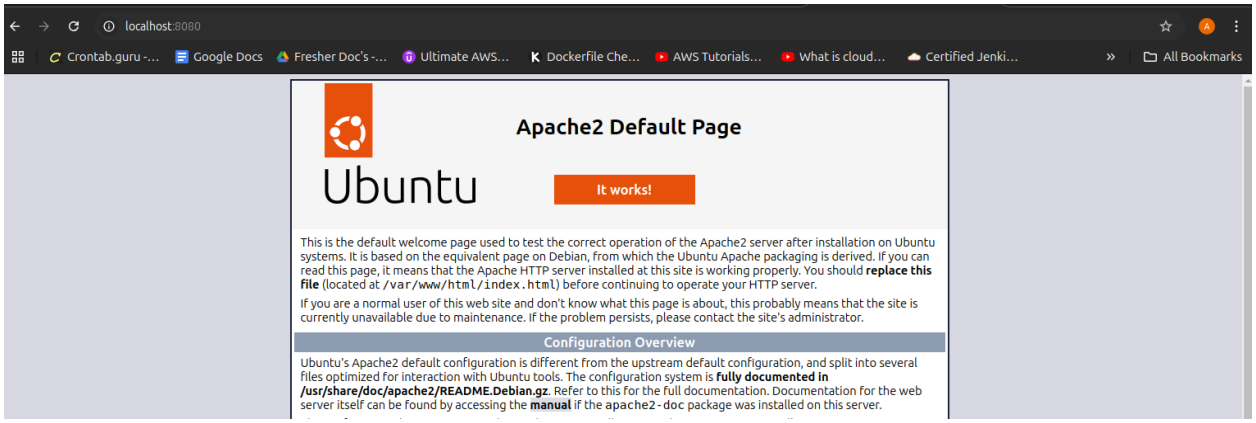
```
sudo ss -tuln | grep 8080
```

```
sudo ss -tuln | grep 80
```

```
alfiya@alfiya:~$ sudo ss -tuln | grep 8080
sudo ss -tuln | grep 80
tcp   LISTEN 0      511                                     *:8080            *:*
udp   UNCONN 0      0      [fe80::fe6b:fe52:7445:7ca9]%wlp1s0:546          [::]:*
tcp   LISTEN 0      511                               0.0.0.0:80        0.0.0.0:*
tcp   LISTEN 0      511                                     *:8080            *:*
alfiya@alfiya:~$
```

What This Confirms:

- Nginx is running as a reverse proxy on port 80 (standard HTTP port).
- Apache is running on port 8080 (backend service for Nginx).
- This setup avoids port conflicts and allows both services to coexist and work together.




```
alfiya@alfiya:~$ curl -I http://testnginx.com
HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Sun, 01 Dec 2024 02:16:55 GMT
Content-Type: text/html
Content-Length: 10671
Connection: keep-alive
Last-Modified: Sun, 01 Dec 2024 01:12:47 GMT
ETag: "29af-6282b214fa44d"
Accept-Ranges: bytes
Vary: Accept-Encoding

alfiya@alfiya:~$ ping testnginx.com
PING testnginx.com (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.021 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.068 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.068 ms
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.045 ms
64 bytes from localhost (127.0.0.1): icmp_seq=5 ttl=64 time=0.065 ms
64 bytes from localhost (127.0.0.1): icmp_seq=6 ttl=64 time=0.048 ms
64 bytes from localhost (127.0.0.1): icmp_seq=7 ttl=64 time=0.093 ms
64 bytes from localhost (127.0.0.1): icmp_seq=8 ttl=64 time=0.047 ms
64 bytes from localhost (127.0.0.1): icmp_seq=9 ttl=64 time=0.131 ms
```

## 2. User Management

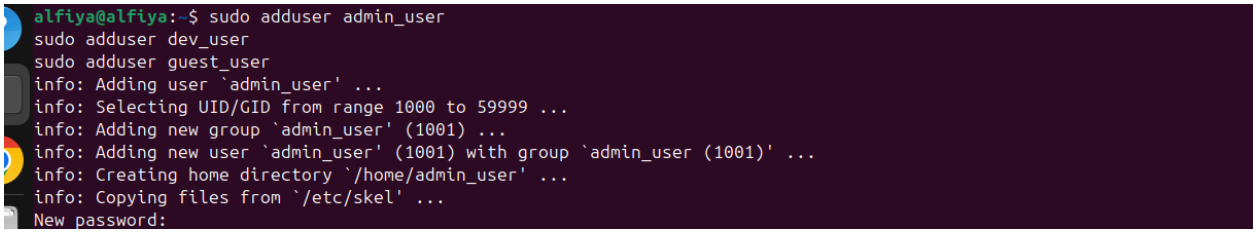
- Objective: Create and manage user roles with appropriate permissions.
- Create users for admins, developers, and guests.
- Assign proper permissions to directories /var/www/html for web admins.

Steps:

1. Add users:

```
sudo adduser admin_user
```

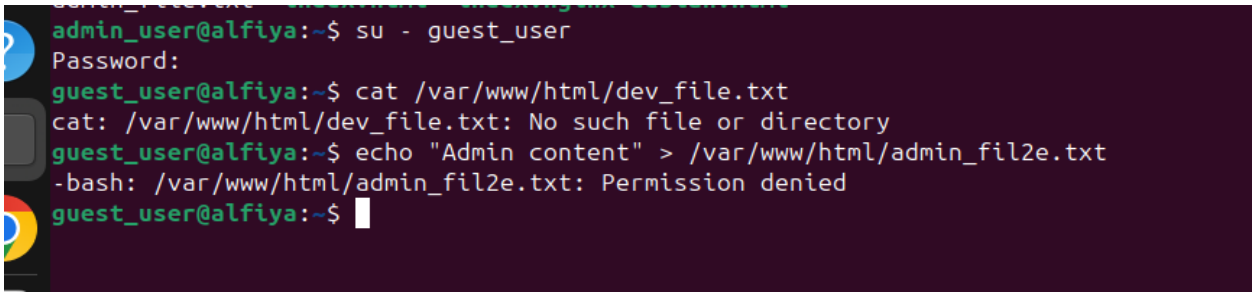
```
sudo adduser dev_user
sudo adduser guest_user
```




2. Set permissions:

```
sudo chown -R admin_user:www-data /var/www/html
sudo chmod -R 775 /var/www/html
```


3. Access Examples
    - Admin User Access: Can create, modify, or delete files in `/var/www/html` since they are the owner.Full control over the directory and its contents.
    - Developer Access: Can edit or create files in `/var/www/html` because they belong to the `www-data` group. Cannot delete files not created by them unless explicitly allowed.
    - Guest User Access: Can only read and execute files in `/var/www/html`.Cannot modify, create, or delete files due to limited permissions.

3. Database Management: Creating a Simple Website with Form Submission and Database Integration.

Steps:

1. Install Required Software and ensure Apache or Nginx, PHP and MySQL are installed.

```
sudo apt update
sudo apt install apache2 mysql-server php php-mysql -y
```

```
alfiya@alfiya:~$ sudo apt install mysql-server php php-mysql -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libaio1t64 libapache2-mod-php8.3 libcgi-fast-perl libcgi-pm-perl
  libevent-core-2.1-7t64 libevent-pthreads-2.1-7t64 libfcgi-bin libfcgi-perl
  libfcgi0t64 libhtml-template-perl libmecab2 libprotobuf-lite32t64
```

2. Configure MySQL Database: Log in to MySQL:

```
sudo mysql -u root -p
```

```
alfiya@alfiya:~$
sudo mysql -u root -p

Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.40-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql>
```

3. Create a new database and create a user and grant user permissions
4. Use the database and create a table for storing form data

```
CREATE DATABASE formdb;
CREATE USER 'formuser'@'localhost' IDENTIFIED BY 'securepassword';
GRANT ALL PRIVILEGES ON formdb.* TO 'formuser'@'localhost';
FLUSH PRIVILEGES;
USE formdb;
CREATE TABLE submissions (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
```

```
    email VARCHAR(255) NOT NULL,
    submitted_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
mysql> CREATE DATABASE formdb;
Query OK, 1 row affected (0.01 sec)

mysql>
mysql>
```

```
mysql> CREATE USER 'formuser'@'localhost' IDENTIFIED BY 'securepassword';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON formdb.* TO 'formuser'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)

mysql> USE formdb;
Database changed
mysql> CREATE TABLE submissions (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(255) NOT NULL,
    ->     email VARCHAR(255) NOT NULL,
    ->     submitted_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql>
```

5. Create the HTML Form
    - Place the form in Apache's web root directory /var/www/html.
    - Create a file named form.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Submit Your Info</title>
</head>
<body>
    <h1>Submit Your Information</h1>
    <form action="submit.php" method="POST">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required><br><br>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required><br><br>
```

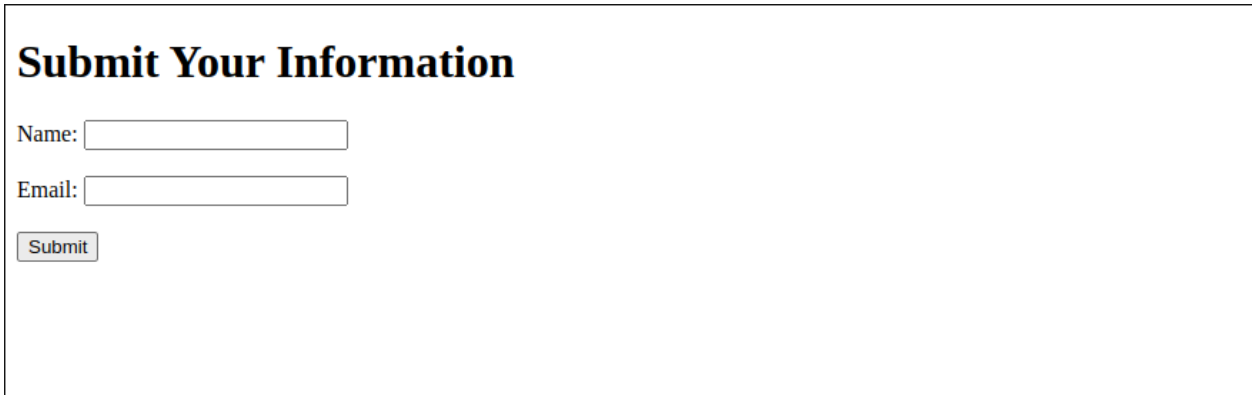
```
    <button type="submit">Submit</button>
  </form>
</body>
</html>
```

6. Create the PHP Backend
    - Create a file named submit.php in the same directory

```
<?php
// Database connection details
$servername = "localhost";
$username = "formuser";
$password = "securepassword";
$dbname = "formdb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// Get form data
$name = $_POST['name'];
$email = $_POST['email'];

// Insert data into database
$sql = "INSERT INTO submissions (name, email) VALUES ('$name', '$email')";

if ($conn->query($sql) === TRUE) {
  echo "Data submitted successfully!";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}

// Close connection
$conn->close();
?>
```

7. Test the Website
    - Open a browser and navigate to the form
    - Fill out the form and submit it.
    - Verify that the data is stored in the database

```
sudo mysql -u root -p
USE formdb;
SELECT * FROM submissions;
```




```
alfiya@alfiya:~$ sudo mysql -u root -p
[sudo] password for alfiya:
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.40-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> USE formdb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * FROM submissions;
+----+------+---------------+---------------------+
| id | name | email         | submitted_at        |
+----+------+---------------+---------------------+
|  1 | demo | demo@gmail.com | 2024-12-02 17:11:13 |
+----+------+---------------+---------------------+
1 row in set (0.00 sec)

mysql>
```

4. Automates backup of website data that is form , configuration files, and databases and further compressing and archiving it using shell scripts.

Steps:

1. Create the Script File
    - Open a new file
    - sudo nano backup_website.sh
    - Backup MySQL database

```
# Variables
DB_USER="formuser"
DB_PASS="securepassword"
DB_NAME="formdb"
BACKUP_DIR="backups"
DATE=$(date +'%Y-%m-%d_%H-%M-%S')
APACHE_CONF="/etc/apache2/sites-available"
NGINX_CONF="/etc/nginx/sites-available"
WEBSITE_DATA="/var/www"

# Create backup directory if not exists
if [ ! -d "$BACKUP_DIR" ]; then
    echo "Creating backup directory $BACKUP_DIR..."
    mkdir -p "$BACKUP_DIR"
    chmod 755 "$BACKUP_DIR"
fi

DIR/$DB_NAME-$DATE.sql"
if [ $? -eq 0 ]; then
    echo "Database backup successful!"
else
    echo "Database backup failed!" >&2
    exit 1
fi

# Backup Apache2 configuration
echo "Backing up Apache2 configuration files..."
tar -czf "$BACKUP_DIR/apache2-config-$DATE.tar.gz" "$APACHE_CONF"
if [ $? -eq 0 ]; then
    echo "Apache2 configuration backup successful!"
else
```

```
    echo "Apache2 configuration backup failed!" >&2
    exit 1
fi

# Backup Nginx configuration
echo "Backing up Nginx configuration files..."
tar -czf "$BACKUP_DIR/nginx-config-$DATE.tar.gz" "$NGINX_CONF"
if [ $? -eq 0 ]; then
    echo "Nginx configuration backup successful!"
else
    echo "Nginx configuration backup failed!" >&2
    exit 1
fi

# Backup website data
echo "Backing up website data..."
tar -czf "$BACKUP_DIR/website-data-$DATE.tar.gz" "$WEBSITE_DATA"
if [ $? -eq 0 ]; then
    echo "Website data backup successful!"
else
    echo "Website data backup failed!" >&2
    exit 1
fi

# Create a final archive
echo "Creating a final backup archive..."
tar -czf "$BACKUP_DIR/full-backup-$DATE.tar.gz"
"$BACKUP_DIR"/*.tar.gz "$BACKUP_DIR"/*.sql"
if [ $? -eq 0 ]; then
    echo "Final archive created successfully!"
    #rm -f "$BACKUP_DIR"/*.tar.gz "$BACKUP_DIR"/*.sql
else
    echo "Failed to create final archive!" >&2
    exit 1
Fi

echo "Backup completed! All backups are stored in $BACKUP_DIR.”
```

- Save and Exit

2. Make the Script Executable:

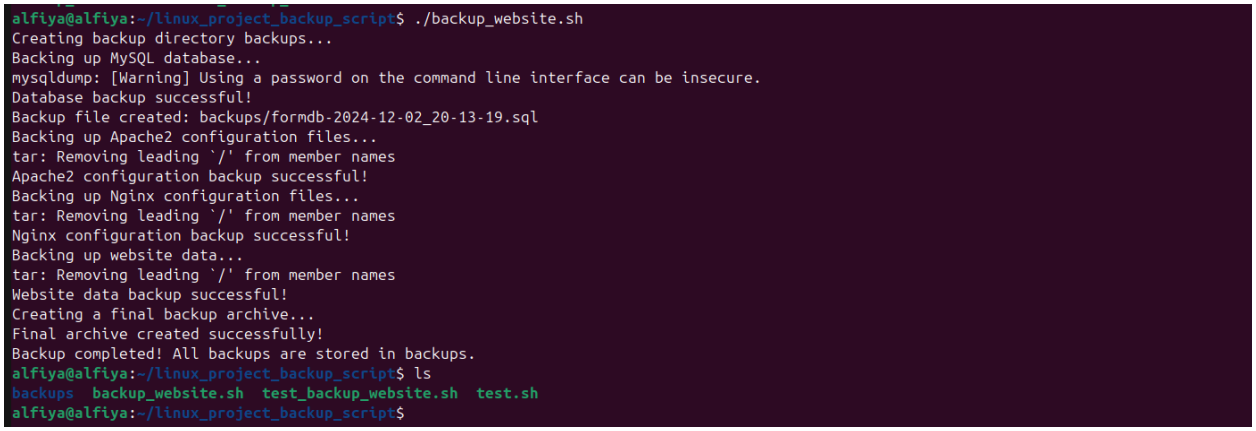
```
chmod +x full_backup.sh
```

3. Execute the Script

```
./backup_website.sh
```

4. Verify the Backup
5. Check the backups directory

```
ls backups
```




(Note: Use the name Username, password, DB and Table also check if the user has privileges for taking backups)

5. Automating backups using cron jobs.

1. Open Crontab for Editing You need to set up a cron job that will automatically run the backup script on a regular basis (e.g., daily, weekly, etc.).
2. Run the following command to edit the cron jobs:

```
crontab -e
```

3. Add a Cron Job To automate the backup process, add a line in the crontab file for the desired schedule. For example, to run the backup every day at 2 AM, add this line:

```
0 2 * * * /path/to/backup_website.sh
```

4. Verify Cron Job
    - To verify that the cron job has been added correctly:
    - List the active cron jobs:

```
crontab -l
```

- Check that the script is being executed as expected by looking files