

Wydział Elektrotechniki i Informatyki

Katedra Informatyki



Programowanie Full-Stack w Chmurze Obliczeniowej

***Implementacja wybranego stack-a wraz z przykładową aplikacją –
architektura i konfiguracja stack-a, sposób wdrożenia oraz użycia przez
developera. LAMP stack***

Autor:

Michał Grzeszuk

Lublin 04.06.2024 r.

Spis treści

Wydział Elektrotechniki i Informatyki	1
1. Czym jest LAMP stack	3
2. Zastosowanie LAMP stack	4
1. Tworzenie stron internetowych i blogów	4
2. Systemy zarządzania treścią (CMS)	4
3. Aplikacje e-commerce	4
4. Forum internetowe	4
3. Instalacja LAMP stack	5
4. Opis implementacji	24
5. Podsumowanie	25

1. Czym jest LAMP stack

LAMP stack to akronim oznaczający zestaw oprogramowania wykorzystywanego do tworzenia dynamicznych stron internetowych i aplikacji. Skrót ten pochodzi od pierwszych liter czterech kluczowych komponentów, które tworzą LAMP stack:

1. **Linux:** System operacyjny na którym działają pozostałe elementy stosu. Linux jest znany ze swojej stabilności, bezpieczeństwa i otwartego kodu źródłowego.
2. **Apache:** Serwer HTTP, który obsługuje żądania przeglądarek internetowych i dostarcza strony internetowe do użytkowników. Apache jest jednym z najpopularniejszych serwerów WWW na świecie.
3. **MySQL:** System zarządzania relacyjnymi bazami danych (RDBMS), który przechowuje dane aplikacji. MySQL jest również oprogramowaniem open source i jest szeroko stosowany w różnych projektach.
4. **PHP** (czasami Perl lub Python): Język skryptowy stosowany do tworzenia dynamicznych treści stron internetowych. PHP działa na serwerze, wykonując kod, który generuje HTML wysyłany do przeglądarki użytkownika.

Podsumowując, LAMP stack to kompletny zestaw narzędzi programistycznych umożliwiających tworzenie, testowanie i wdrażanie aplikacji internetowych. Każdy z komponentów pełni określoną rolę, a razem tworzą solidną podstawę dla rozwoju nowoczesnych aplikacji webowych.

2. Zastosowanie LAMP stack

1. Tworzenie stron internetowych i blogów

LAMP stack jest popularnym wyborem do tworzenia i hostowania dynamicznych stron internetowych oraz blogów. Dzięki PHP można tworzyć dynamiczne treści, a MySQL przechowuje dane użytkowników i treści stron.

- **Przykład:** WordPress, Joomla, Drupal.

2. Systemy zarządzania treścią (CMS)

CMS to aplikacje, które umożliwiają łatwe tworzenie, edytowanie i zarządzanie treściami na stronie internetowej bez konieczności posiadania wiedzy programistycznej.

- **Przykład:** WordPress jest najbardziej znanym CMS opartym na LAMP stack.

3. Aplikacje e-commerce

LAMP stack jest również wykorzystywany do tworzenia platform e-commerce, które umożliwiają handel online. Dzięki niemu można zarządzać produktami, zamówieniami, płatnościami i danymi klientów.

- **Przykład:** Magento, PrestaShop, OpenCart.

4. Forum internetowe

Fora internetowe są popularnymi miejscami do dyskusji online, gdzie użytkownicy mogą tworzyć posty, odpowiadać na nie i uczestniczyć w społeczności.

- **Przykład:** phpBB, MyBB, SMF (Simple Machines Forum).

3. Instalacja LAMP stack

- Aktualizacja systemu:

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains two lines of text: 'sudo apt update' and 'sudo apt upgrade'.

```
sudo apt update  
sudo apt upgrade
```

- Instalacja Apache

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains two lines of text: 'sudo apt install apache2' and 'sudo systemctl status apache2'.

```
sudo apt install apache2  
sudo systemctl status apache2
```

```

michal@DESKTOP-T2TJMGJ:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap mailcap mime-support
  ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser bzip2-doc
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap mailcap mime-support
  ssl-cert
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 1999 kB of archives.
After this operation, 8011 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libapr1 amd64 1.7.0-Subuntu0.22.04.1 [108 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libaprutil1 amd64 1.6.1-Subuntu4.22.04.2 [92.8 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.1-Subuntu4.22.04.2 [11.3 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libaprutil1-ldap amd64 1.6.1-Subuntu4.22.04.2 [9170 B]
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 apache2-bin amd64 2.4.52-1ubuntu4.9 [1347 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 apache2-data all 2.4.52-1ubuntu4.9 [165 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 apache2-utils amd64 2.4.52-1ubuntu4.9 [88.7 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 mailcap all 3.70+nmulubuntu1 [23.8 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/main amd64 mime-support all 3.66 [3696 B]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 apache2 amd64 2.4.52-1ubuntu4.9 [97.9 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy/main amd64 bzip2 amd64 1.0.8-5build1 [34.8 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/main amd64 ssl-cert all 1.1.2 [17.4 kB]
Fetched 1999 kB in 1s (1477 kB/s)
Preconfiguring packages ...

```

```

michal@michal-Vostro-5568:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-06-13 10:59:44 CEST; 19s ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 28610 (apache2)
       Tasks: 55 (limit: 14852)
      Memory: 5.7M (peak: 6.3M)
         CPU: 137ms
    CGroup: /system.slice/apache2.service
            └─28610 /usr/sbin/apache2 -k start
              └─28612 /usr/sbin/apache2 -k start
                └─28613 /usr/sbin/apache2 -k start

Jun 13 10:59:44 michal-Vostro-5568 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jun 13 10:59:44 michal-Vostro-5568 apache2[28609]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive
Jun 13 10:59:44 michal-Vostro-5568 systemd[1]: Started apache2.service - The Apache HTTP Server.
(lines 1-16/16) (END)

```

- Instalacja MySQL

```

sudo apt install mysql-server
sudo mysql_secure_installation
sudo mysql -u root -p

```

```

michal@DESKTOP-T2TJMG3:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl
  libfcgi0ldbl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils
  mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server-8.0 mysql-server-core-8.0
Suggested packages:
  libdata-dump-perl libipc-sharedcache-perl libbusiness-isbn-perl libwww-perl mailx tinymce
The following NEW packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl
  libfcgi0ldbl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils
  mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server mysql-server-core-8.0
0 upgraded, 29 newly installed, 0 to remove and 0 not upgraded.
Need to get 29.6 MB of archives.
After this operation, 243 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 mysql-common all 5.8+1.0.8 [7212 B]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-core-8.0 amd64 8.0.37-0ubuntu0.22.04.3 [2762 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-8.0 amd64 8.0.37-0ubuntu0.22.04.3 [22.7 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/main amd64 libaio1 amd64 0.3.112-13build1 [7176 B]

```

```

michal@michal-Vostro-5568:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.37-0ubuntu0.24.04.1 (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> alter user 'root'@'localhost' identified with mysql_native_password by '5520';
Query OK, 0 rows affected (0.02 sec)

mysql> flush privileges;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'privileges' at line 1
mysql> flush privileges;
Query OK, 0 rows affected (0.03 sec)

mysql> exit;

```

```

STRONG Length >= 8, numeric, mixed case, special characters and dictionary file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 0
Using existing password for root.

Estimated strength of the password: 25
Change the password for root ? ((Press y|Y for Yes, any other key for No) : n

... skipping.
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : n

... skipping.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : n

... skipping.
By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : n

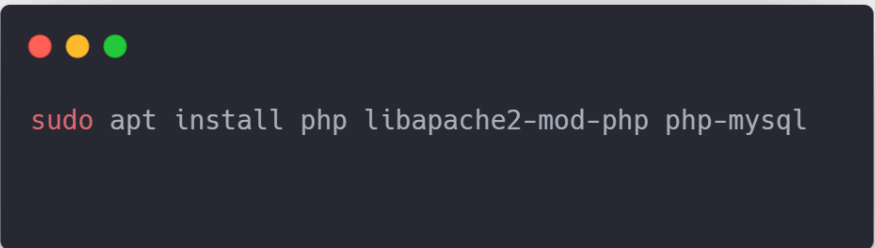
... skipping.
Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : n

... skipping.
All done!
michal@michal-Vostro-5568:~$

```

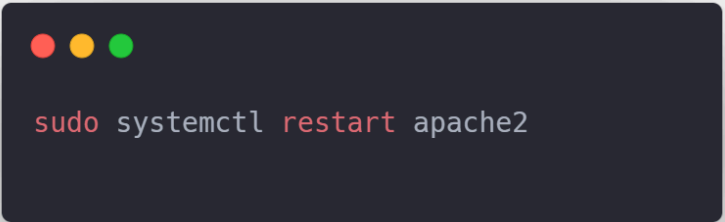
- Instalacja PHP



```
sudo apt install php libapache2-mod-php php-mysql
```

```
Michał@Michał-Vostro-5568:~$ php -v
PHP 8.3.6 (cli) (built: Apr 15 2024 19:21:47) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.3.6, Copyright (c) Zend Technologies
with Zend OPcache v8.3.6, Copyright (c), by Zend Technologies
Michał@Michał-Vostro-5568:~$
```

- Restart Apache



```
sudo systemctl restart apache2
```

```
Michał@Michał-Vostro-5568:~$ sudo systemctl restart apache2
Michał@Michał-Vostro-5568:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-06-13 11:09:35 CEST; 8s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 30423 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 30426 (apache2)
     Tasks: 6 (limit: 14052)
    Memory: 10.9M (peak: 11.4M)
       CPU: 203ms
   CGroup: /system.slice/apache2.service
           └─30426 /usr/sbin/apache2 -k start
             └─30428 /usr/sbin/apache2 -k start
               └─30429 /usr/sbin/apache2 -k start
                 └─30430 /usr/sbin/apache2 -k start
                   └─30431 /usr/sbin/apache2 -k start
                     └─30432 /usr/sbin/apache2 -k start

Jun 13 11:09:35 michal-Vostro-5568 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jun 13 11:09:35 michal-Vostro-5568 apachectl[30425]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive
Jun 13 11:09:35 michal-Vostro-5568 systemd[1]: Started apache2.service - The Apache HTTP Server.
[lines 1-20/20 (END)]
```


- Konfiguracja MySQL

```
CREATE DATABASE myapp;

CREATE USER 'myuser' @ 'localhost' IDENTIFIED BY
'mypassword';

GRANT ALL PRIVILEGES ON myapp.* TO 'myuser' @
'localhost';

FLUSH PRIVILEGES;
```

```
CREATE DATABASE LampDb;

USE LampDb;

CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    surname VARCHAR(255) NOT NULL,
    age INT NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE
);

CREATE TABLE profiles (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    bio TEXT,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
    CASCADE
```

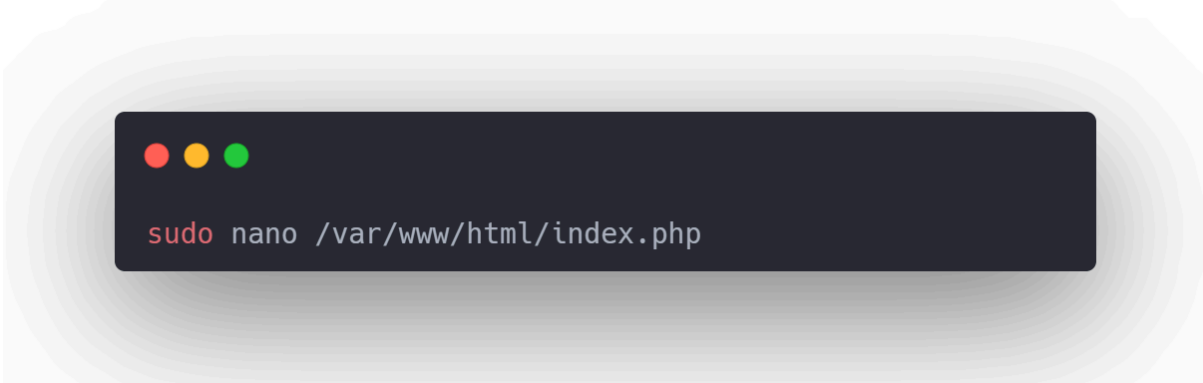
```
mysql> select User, Host From mysql.user;
+-----+-----+
| User          | Host          |
+-----+-----+
| debian-sys-maint | localhost    |
| mysql.infoschema | localhost    |
| mysql.session   | localhost    |
| mysql.sys       | localhost    |
| root           | localhost    |
| grzeszuk        | lolcahost    |
+-----+-----+
6 rows in set (0.00 sec)

mysql>
```

```
root@micah-Vostro-5568:/var/www/html/css# systemctl status mysql
● mysql.service - MySQL Community Server
   Loaded: loaded (/usr/lib/systemd/system/mysql.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-06-13 11:02:36 CEST; 22min ago
     Main PID: 22292 (mysqld)
    Status: "Server is operational"
       Tasks: 38 (limit: 14052)
      Memory: 367.9M (peak: 379.3M)
         CPU: 17.674s
    CGroup: /system.slice/mysql.service
            └─22292 /usr/sbin/mysqld

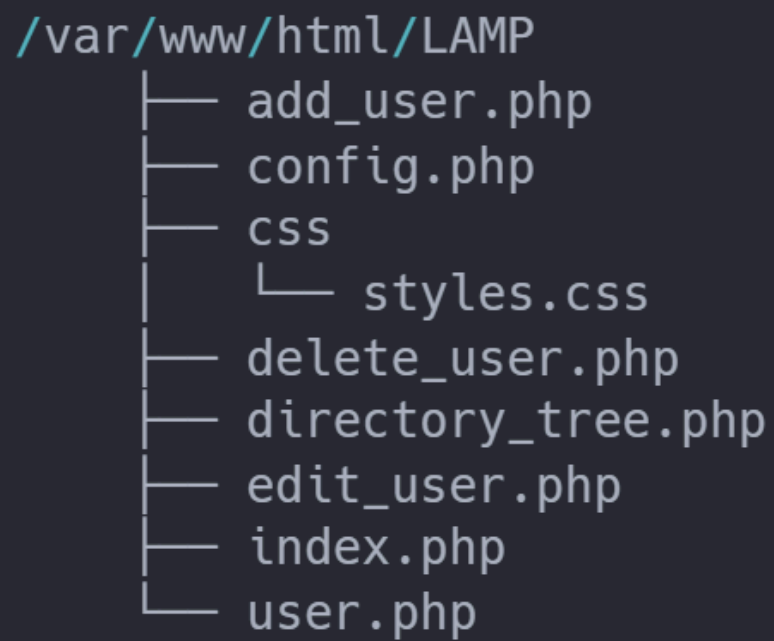
Jun 13 11:02:34 micah-Vostro-5568 systemd[1]: Starting mysql.service - MySQL Community Server...
Jun 13 11:02:36 micah-Vostro-5568 systemd[1]: Started mysql.service - MySQL Community Server.
```

- Tworzenie przykładowej aplikacji PHP



```
sudo nano /var/www/html/index.php
```

- Struktura katalogów

A terminal window with a dark background and three colored window control buttons (red, yellow, green) at the top left. It displays a directory listing for the path /var/www/html/LAMP. The listing shows a vertical line of files and directories, with horizontal lines branching off to the right to indicate sub-items. The files are: add_user.php, config.php, css (which contains styles.css), delete_user.php, directory_tree.php, edit_user.php, index.php, and user.php.

```
/var/www/html/LAMP
├── add_user.php
├── config.php
├── css
│   └── styles.css
├── delete_user.php
├── directory_tree.php
├── edit_user.php
├── index.php
└── user.php
```

- index.php

```
<?php
include 'config.php';

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

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

$sql = "SELECT id, name, surname, age, email FROM users";
$result = $conn->query($sql);
?>

<!DOCTYPE html>
<html>
<head>
    <title>MyApp</title>
    <link rel="stylesheet" type="text/css"
href="css/styles.css">
</head>
<body>
    <h1>User List</h1>
    <table>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Surname</th>
            <th>Age</th>
            <th>Email</th>
            <th>Actions</th>
        </tr>
        <?php
        if ($result->num_rows > 0) {
            while($row = $result->fetch_assoc()) {
                echo "<tr><td>" . $row["id"]. "</td>
                <td><a href='user.php?id=" . $row["id"].
                ">" . $row["name"]. "</a></td>
                <td>" . $row["surname"]. "</td>
                <td>" . $row["age"]. "</td>
                <td>" . $row["email"]. "</td>
                <td>
                    <a href='edit_user.php?id=" .
                    $row["id"]. "'>Edit</a> |
                    <a href='delete_user.php?id=" .
                    $row["id"]. "' onclick=\\"return confirm('Are you sure you want
                    to delete this user?');\\">Delete</a>
                </td>
            </tr>";
            }
        } else {
            echo "<tr><td colspan='6'>0 results</td></tr>";
        }
        $conn->close();
    ?>
    </table>
    <br>
    <a href="add_user.php">Add New User</a>
</body>
</html>
```

- add_user.php

```
<?php
include 'config.php';

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST['name'];
    $surname = $_POST['surname'];
    $age = $_POST['age'];
    $email = $_POST['email'];
    $bio = $_POST['bio'];

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

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

    $sql_user = "INSERT INTO users (name, surname, age, email) VALUES ('$name', '$surname', '$age', '$email')";
    if ($conn->query($sql_user) === TRUE) {
        $user_id = $conn->insert_id;
        $sql_profile = "INSERT INTO profiles (user_id, bio) VALUES ('$user_id', '$bio')";
        if ($conn->query($sql_profile) === TRUE) {
            echo "New user and profile created successfully";
        } else {
            echo "Error: " . $sql_profile . "<br>" . $conn->error;
        }
    } else {
        echo "Error: " . $sql_user . "<br>" . $conn->error;
    }
}

$conn->close();
?>

<!DOCTYPE html>
<html>
<head>
    <title>Add User</title>
    <link rel="stylesheet" type="text/css" href="css/styles.css">
</head>
<body>
    <h1>Add New User</h1>
    <form method="post" action="add_user.php">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required>
        <br><br>
        <label for="surname">Surname:</label>
        <input type="text" id="surname" name="surname" required>
        <br><br>
        <label for="age">Age:</label>
        <input type="number" id="age" name="age" required>
        <br><br>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required>
        <br><br>
        <label for="bio">Bio:</label>
        <textarea id="bio" name="bio"></textarea>
        <br><br>
        <input type="submit" value="Add User">
    </form>
    <br>
    <a href="index.php">Back to User List</a>
</body>
</html>
```

- config.php

```
<?php
$servername = "localhost";
$username = "root";
$password = "your_password";
$dbname = "your_database";
?>
```

- delete_user.php

```
<?php
include 'config.php';

if (isset($_GET['id'])) {
    $id = intval($_GET['id']);

    // Połączenie z bazą danych
    $conn = new mysqli($servername, $username, $password,
$dbname);
    // Sprawdzenie połączenia
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    // Usunięcie użytkownika
    $sql = "DELETE FROM users WHERE id = $id";
    if ($conn->query($sql) === TRUE) {
        echo "User deleted successfully";
    } else {
        echo "Error deleting user: " . $conn->error;
    }

    $conn->close();

    // Przekierowanie z powrotem do listy użytkowników
    header("Location: index.php");
    exit();
} else {
    echo "No user ID specified.";
}
?>
```

- edit_user.php

```
<?php
include 'config.php';

if (isset($_GET['id'])) {
    $id = intval($_GET['id']);

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

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

    $sql = "SELECT * FROM users WHERE id = $id";
    $result = $conn->query($sql);
    $row = $result->fetch_assoc();

    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $name = $_POST['name'];
        $surname = $_POST['surname'];
        $age = $_POST['age'];
        $email = $_POST['email'];
        $bio = $_POST['bio'];

        $sql = "UPDATE users SET name='$name', surname='$surname', age='$age', email='$email' WHERE
id=$id";
        if ($conn->query($sql) === TRUE) {
            $sql_profile = "UPDATE profiles SET bio='$bio' WHERE user_id=$id";
            if ($conn->query($sql_profile) === TRUE) {
                echo "User updated successfully";
            } else {
                echo "Error updating profile: " . $conn->error;
            }
        } else {
            echo "Error updating user: " . $conn->error;
        }

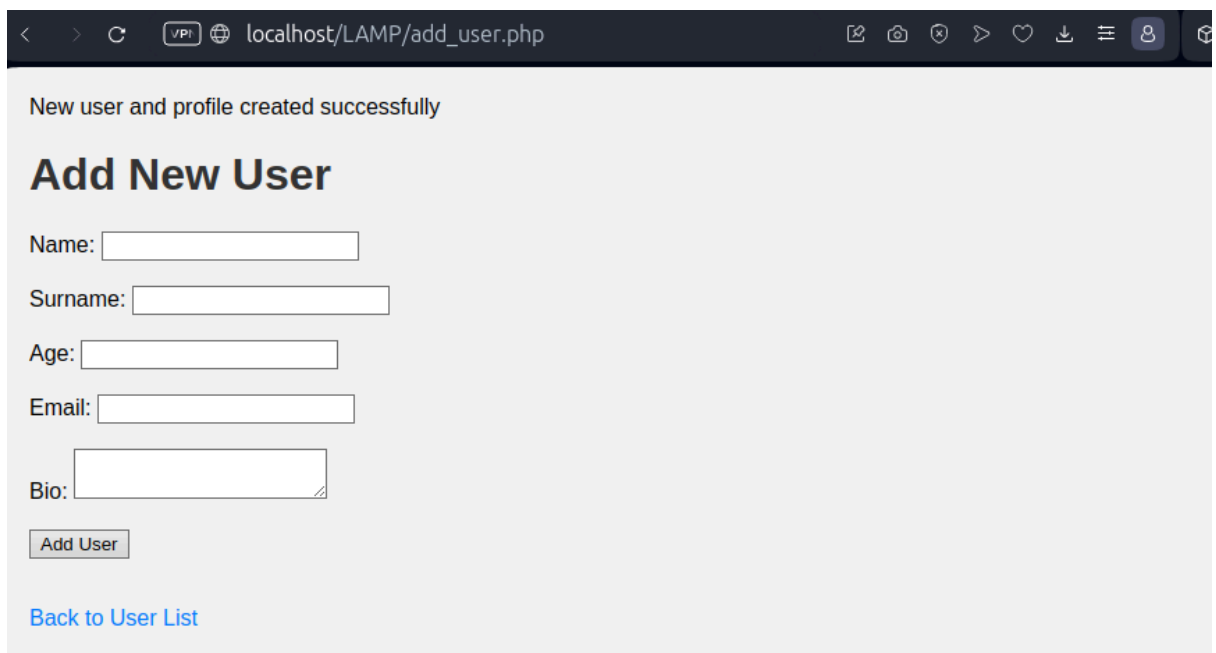
        $conn->close();
        header("Location: index.php");
        exit();
    }
} else {
    echo "No user ID specified.";
}
?>

<!DOCTYPE html>
<html>
<head>
    <title>Edit User</title>
    <link rel="stylesheet" type="text/css" href="css/styles.css">
</head>
<body>
    <h1>Edit User</h1>
    <form method="post" action="edit_user.php?id=<?php echo $id; ?>">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" value="<?php echo $row['name']; ?>" required>
        <br><br>
        <label for="surname">Surname:</label>
        <input type="text" id="surname" name="surname" value="<?php echo $row['surname']; ?>" required>
        <br><br>
        <label for="age">Age:</label>
        <input type="number" id="age" name="age" value="<?php echo $row['age']; ?>" required>
        <br><br>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" value="<?php echo $row['email']; ?>" required>
        <br><br>
        <label for="bio">Bio:</label>
        <textarea id="bio" name="bio"><?php
            $sql_profile = "SELECT bio FROM profiles WHERE user_id = $id";
            $result_profile = $conn->query($sql_profile);
            $row_profile = $result_profile->fetch_assoc();
            echo $row_profile['bio'];
        ?></textarea>
        <br><br>
        <input type="submit" value="Update User">
    </form>
    <br>
    <a href="index.php">Back to User List</a>
</body>
</html>
```

- **Widoki aplikacji**



Lista użytkowników



Dodawanie nowego użytkownika

localhost/LAMP/edit_user.php

Edit User

Name:

Surname:

Age:

Email:

Bio:

[Back to User List](#)

Edycja użytkownika

localhost/LAMP/index.php

User List

localhost says
Are you sure you want to delete this user?

Cancel OK

ID	Name	Surname	Age	Email	Actions
2	Tomek	Nowak	23	t@gmail.com	Edit Delete

[Add New User](#)

Usuwanie użytkownika

- Struktura plików projektu

```
root@michal-Vostro-5568:/var/www/html/LAMP# tree
.
├── add_user.php
├── config.php
├── css
│   └── styles.css
├── delete_user.php
├── edit_user.php
├── index.html
├── index.php
└── user.php
```

- Tworzenie obrazu Docker aplikacji

```
FROM php:7.4-apache

# Skopiuj pliki aplikacji do katalogu domyślnego serwera Apache
COPY . /var/www/html/

# Ustaw właściwe uprawnienia
RUN chown -R www-data:www-data /var/www/html

# Włącz moduł Apache mod_rewrite
RUN a2enmod rewrite

# Expose port 80
EXPOSE 80
```

```
root@micah-Vostro-5568:/var/www/html/LAMP# docker build -t my-php-app .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 28.16kB
Step 1/5 : FROM php:7.4-apache
7.4-apache: Pulling from library/php
a603fa5e3b41: Pull complete
c428f1a49423: Pull complete
156740b07ef8: Pull complete
fb5a4c8af82f: Pull complete
25f85b498fd5: Pull complete
9b233e420ac7: Pull complete
fe42347c4ecf: Pull complete
d14eb2ed1e17: Pull complete
66d98f73acb6: Pull complete
d2c43c5efbc8: Pull complete
ab590b48ea47: Pull complete
80692ae2d067: Pull complete
05e465aaa99a: Pull complete
Digest: sha256:c9d7e608f73832673479770d66aacc8100011ec751d1905ff63fae3fe2e0ca6d
Status: Downloaded newer image for php:7.4-apache
--> 20a3732f422b
Step 2/5 : COPY . /var/www/html/
--> 91504c027fb6
Step 3/5 : RUN chown -R www-data:www-data /var/www/html
--> Running in 0bc7afddc556
Removing intermediate container 0bc7afddc556
--> 9727e914cf16
Step 4/5 : RUN a2enmod rewrite
--> Running in 99a6f77dffd7
Enabling module rewrite.
To activate the new configuration, you need to run:
service apache2 restart
Removing intermediate container 99a6f77dffd7
--> 673e990b8c55
Step 5/5 : EXPOSE 80
--> Running in 0873e6a9d921
Removing intermediate container 0873e6a9d921
--> ade59b241c0f
Successfully built ade59b241c0f
Successfully tagged my-php-app:latest
root@micah-Vostro-5568:/var/www/html/LAMP#
```

- Deployment aplikacji

deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-php-app
  labels:
    app: my-php-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-php-app
  template:
    metadata:
      labels:
        app: my-php-app
    spec:
      containers:
        - name: php-apache
          image: my-php-app
          ports:
            - containerPort: 80
          volumeMounts:
            - name: app-config
              mountPath: /var/www/html/config.php
              subPath: config.php
      volumes:
        - name: app-config
          configMap:
            name: app-config
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  config.php: |
    <?php
    $servername = "mysql-service";
    $username = "root";
    $password = "your_password";
    $dbname = "LampDb";
    ?>
---
apiVersion: v1
kind: Service
metadata:
  name: my-php-app-service
spec:
  selector:
    app: my-php-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: LoadBalancer
```

mysql-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:5.7
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: "your_password"
            - name: MYSQL_DATABASE
              value: "LampDb"
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mysql-storage
              mountPath: /var/lib/mysql
      volumes:
        - name: mysql-storage
          persistentVolumeClaim:
            claimName: mysql-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: mysql-service
spec:
  ports:
    - port: 3306
  selector:
    app: mysql
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

```

minikube v1.33.1 on Ubuntu 24.04
✨ Using the docker driver based on user configuration
❗ The "docker" driver should not be used with root privileges. If you wish to continue as root, use --force.
💡 If you are running minikube within a VM, consider using --driver=none:
   https://minikube.sigs.k8s.io/docs/reference/drivers/none/

❌ Exiting due to DRV_AS_ROOT: The "docker" driver should not be used with root privileges

root@micah-Vostro-5568:/var/www/html/LAMP# exit
exit
micah@micah-Vostro-5568:/var/www/html$ minikube start
🐤 minikube v1.33.1 on Ubuntu 24.04
✨ Using the docker driver based on existing profile
👍 Starting "minikube" primary control-plane node in "minikube" cluster
📡 Pulling base image v0.0.44 ...
🔄 Restarting existing docker container for "minikube" ...
🔧 Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
🔍 Verifying Kubernetes components...
   ■ Using image registry.k8s.io/ingress-nginx/controller:v1.10.1
   ■ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.1
   ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
   ■ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.1
🔍 Verifying ingress addon...
🌟 Enabled addons: storage-provisioner, default-storageclass, ingress
💡 kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
🚀 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

micah@micah-Vostro-5568:/var/www/html$ ls
LAMP
micah@micah-Vostro-5568:/var/www/html$ cd LAMP/
micah@micah-Vostro-5568:/var/www/html/LAMP$ ls
add_user.php  delete_user.php  edit_user.php  mysql-deployment.yaml
config.php    deployment.yaml  index.html     user.php
css           Dockerfile      index.php
micah@micah-Vostro-5568:/var/www/html/LAMP$ kubectl apply -f mysql-deployment.yaml
deployment.apps/mysql-deployment created
service/mysql-service created
persistentvolumeclaim/mysql-pvc created
micah@micah-Vostro-5568:/var/www/html/LAMP$ kubectl apply -f deployment.yaml
deployment.apps/my-php-app created
configmap/app-config created
service/my-php-app-service created
micah@micah-Vostro-5568:/var/www/html/LAMP$

```

4. Opis implementacji

Podczas instalacji i konfiguracji LAMP stacka oraz wdrażania aplikacji PHP na Kubernetesie wykonaliśmy szereg kroków, które obejmowały instalację poszczególnych komponentów, konfigurację serwera oraz wdrożenie aplikacji. Na początku zainstalowaliśmy system operacyjny Linux, na przykład Ubuntu Server 20.04. Następnie przystąpiliśmy do instalacji serwera Apache za pomocą komendy `sudo apt install apache2`, a po jego uruchomieniu sprawdziliśmy, czy działa poprawnie.

Kolejnym krokiem była instalacja serwera MySQL przy użyciu komendy `sudo apt install mysql-server`. Po instalacji skonfigurowaliśmy MySQL za pomocą `sudo mysql_secure_installation`, co umożliwiło nam zabezpieczenie serwera i ustawienie hasła root. Następnie zainstalowaliśmy PHP oraz niezbędne moduły przy użyciu `sudo apt install php libapache2-mod-php php-mysql`, aby serwer mógł interpretować skrypty PHP i komunikować się z bazą danych.

Stworzyliśmy przykładową aplikację PHP, która łączy się z bazą danych MySQL i wyświetla dane z tabeli `users`. Skrypty PHP umieściliśmy w katalogu `/var/www/html/myapp`, a w pliku `config.php` zdefiniowaliśmy zmienne konfiguracyjne do połączenia z bazą danych. Następnie skonfigurowaliśmy Apache, tworząc plik konfiguracyjny `myapp.conf` i dodając w nim `VirtualHost` wskazujący na naszą aplikację.

Po tych krokach przeprowadziliśmy wdrożenie aplikacji na Kubernetesie. W tym celu stworzyliśmy plik `deployment.yaml`, który definiował deployment aplikacji PHP z dwoma replikami oraz `ConfigMap` przechowujący pliki `index.php` i `config.php`. Dodaliśmy również `Service` typu `LoadBalancer`, aby umożliwić dostęp do aplikacji z zewnątrz, oraz `PersistentVolumeClaim` do przechowywania danych.

Dla bazy danych MySQL stworzyliśmy plik `mysql-deployment.yaml`, który definiował deployment MySQL, `Service` do komunikacji z bazą danych oraz `PersistentVolumeClaim` do przechowywania danych MySQL. Następnie utworzyliśmy namespace dla aplikacji za pomocą `kubectl create namespace myapp` i zastosowaliśmy pliki YAML do Kubernetes za pomocą komend `kubectl apply -f mysql-deployment.yaml -n myapp` oraz `kubectl apply -f deployment.yaml -n myapp`.

Na koniec sprawdziliśmy status wdrożonych podów i usług za pomocą `kubectl get pods -n myapp` i `kubectl get services -n myapp`, aby upewnić się, że wszystkie komponenty działają poprawnie. Użyliśmy zewnętrznego adresu IP wygenerowanego przez `LoadBalancer`, aby uzyskać dostęp do aplikacji.

5. Podsumowanie

LAMP stack, składający się z Linux, Apache, MySQL i PHP, jest popularnym zestawem technologii używanym do budowy i hostowania dynamicznych stron internetowych. Proces instalacji obejmuje zainstalowanie serwera Apache do obsługi HTTP, MySQL jako systemu zarządzania bazą danych oraz PHP jako języka skryptowego do generowania dynamicznych treści. Po zainstalowaniu i skonfigurowaniu tych komponentów, wdrożyliśmy przykładową aplikację PHP, która łączy się z bazą danych MySQL i wyświetla dane na stronie internetowej. Dodatkowo, wdrożyliśmy aplikację na Kubernetesie, korzystając z plików YAML do zdefiniowania deploymentów, usług oraz persistent storage. Cały proces ilustruje, jak zbudować skalowalną i zarządzalną infrastrukturę do hostowania aplikacji internetowych.