Project : Media Streaming with IBM Cloud Video Streaming

Phase 3 : Development part 1

```php
<?php
include_once("libs/database.class.php");
$conn = Database::getConnection();
$sql = "SELECT * FROM `movies`";
$result = $conn->query($sql);
if($result){
    $row = $result->fetch_all(MYSQLI_ASSOC);
}else{
    //do Nothing
}

?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="view-port" content="width=device-width, intial-scale=1.0" >
    <title>Movies.| Home</title>
    <link rel="stylesheet" href="main.css">
</head>
<body>
     <header>
         <nav>
             <a href="" class="logo">Tamilrockers</a>
             <a style="margin-left: 60px;" href="upload.php">Admin</a>

             <div class="mobile">
                 <label class="label" for="toggle">&#9776;</label>
                 <input type="checkbox" id="toggle"/>
                 <ul>
                     <li><a href="upload.php">Admin</a></li>
                 </ul>
             </div>
         </nav>
         <h1>Stream Any Movie at Anytime with No Individual Movie Costs or
Rental Fees. oc laa yea oombalam vaanga</h1>
         <div class="btnGroup">
             <a href="#" class="btnWhite">Yeah</a>
             <div class="whiteBG"></div>
         </div>
         <p>Thousands of movie with <b>no rental fees</b>. Our<br></p>
         <p><b>Forget the theaters</b>! Exclusive access to<br>movies in
theaters on release date.</p>
         <img src="images/ipad.png" alt="" class="ipad">
         <img src="images/iphone.png" alt="" class="iphone">
     </header>
```

```
    <div class="new4KMovies">

        <div class="sectionTitle">
            <h2>NEW 4K MOVIES</h2>
        </div>

       <ul>
       <?php
           foreach($row as $movie){

            echo "<li class='movie'><a  href=$movie[vdeourl]><img src=
$movie[imageurl]></a>
              <button class='btn'>Watch</button></li>";

            } ?>

    </div>
     <div class="callToAction">
         <h1>Watch Your Favorite Movies and Explore Similar Titles. Get
Started Today for <span>Free</span>
     </div>

     <footer>
         <div class="copy">
             <a href="" class="logo">MOVIES.</a>
             <p>&copy; All rights are recived </p>
         </div>
         <ul>
             <li><a href="#">Account</a></li>
             <li><a href="#">Help Center</a></li>
             <li><a href="#">Terms</a></li>
             <li><a href="#">Privacy</a></li>
         </ul>
     </footer>

</body>
</html>
```

Certainly! Let's break down the code in more detail:

1. PHP Opening Tag:

- `<?php ... ?>`: This signifies the beginning of PHP code.

2. Database Connection:

- `include_once("libs/database.class.php");`: This line includes an external PHP file, presumably containing a database connection class or functions.

- `$conn = Database::getConnection();`: It establishes a database connection by calling the `getConnection` method of the `Database` class. The resulting database connection is stored in the `$conn` variable.

3. SQL Query:

- `$sql = "SELECT * FROM movies";`: This line defines an SQL query that selects all columns (`*`) from a table named "movies." The SQL query is stored in the `$sql` variable.

4. Database Query:

- `$result = $conn->query($sql);`: It executes the SQL query using the established database connection (`$conn`). The result of the query is stored in the `$result` variable.

5. Conditional Check:

- `if ($result) { ... }`: This conditional statement checks if the query execution was successful. If it was, the code inside the following block will be executed.

6. Fetching Data:

- `$row = $result->fetch_all(MYSQLI_ASSOC);`: If the query was successful, this line retrieves all the rows from the result set and stores them in the `$row` variable as an array of associative arrays. Each associative array represents a row in the "movies" table.

7. HTML Structure:

- The code then transitions to the HTML portion of the document.

- `<head>` section: It contains metadata, including the character set, title, and a link to an external CSS file for styling.

- `<body>` section: This is the main content of the web page.

8. Header Section:

- The header contains a navigation bar with a logo, links, and a mobile navigation menu.

- It includes a title, introductory text, a button, and images for illustration.

9. New 4K Movies Section:

- This section starts with a title and an unordered list (`<ul>`).

- Inside the `<ul>`, a `foreach` loop iterates through the `$row` array, which holds movie data fetched from the database.

- Within the loop, a list item (`<li>`) is generated for each movie. It includes:

  - An anchor (`<a>`) element linking to the movie's video URL.

  - An image (`<img>`) displaying the movie's poster.

  - A "Watch" button for each movie.

  - If the movie and image is stored in database it will display in order . It will create the template

    For each movie . This is because we use foreach loop

10. Call to Action:

- The "Watch Your Favorite Movies" section includes a call to action and a message encouraging users to get started for free.

11. Footer:

- The footer contains a copyright notice with a logo and a list of links for account, help center, terms, and privacy.

```php
<?php

class Database
{
    public static $conn = null;
    public static function getConnection()
    {
        if (Database::$conn == null) {
            $servername = "localhost";
```

```
            $username = "root" ;
            $password = "";
            $dbname = "movielist";

            // Create connection
            $connection = new mysqli($servername, $username, $password,
$dbname);
            // Check connection
            if ($connection->connect_error) {
                die("Connection failed: " . $connection->connect_error);
//TODO: Replace this with exception handling
            } else {
                //printf("New connection establishing...");
                Database::$conn = $connection; //replacing null with actual
connection
                return Database::$conn;
            }
        } else {
            return Database::$conn;

        }
    }
}
```

This PHP code defines a `Database` class that handles database connections. Here's a breakdown of the code:

1. Class Definition:

- `class Database`: Defines a PHP class named `Database`.

2. Class Properties:

- `public static $conn = null;`: Declares a public static property `$conn` that holds the database connection. It's initially set to `null`.

3. Static Method `getConnection()`:

- `public static function getConnection()`: Defines a static method named `getConnection`. Static methods can be called on the class itself without creating an instance of the class.

4. Database Connection Logic:

- Inside the `getConnection` method, it checks whether a database connection (`$conn`) already exists.

- If a connection doesn't exist (`$conn == null`), it proceeds to create a new database connection.

- It specifies the server name, username, password, and database name for the connection.

- It creates a new `mysqli` database connection using the provided credentials.

- It checks if the connection was successful by inspecting the `connect_error` property of the `mysqli` object.

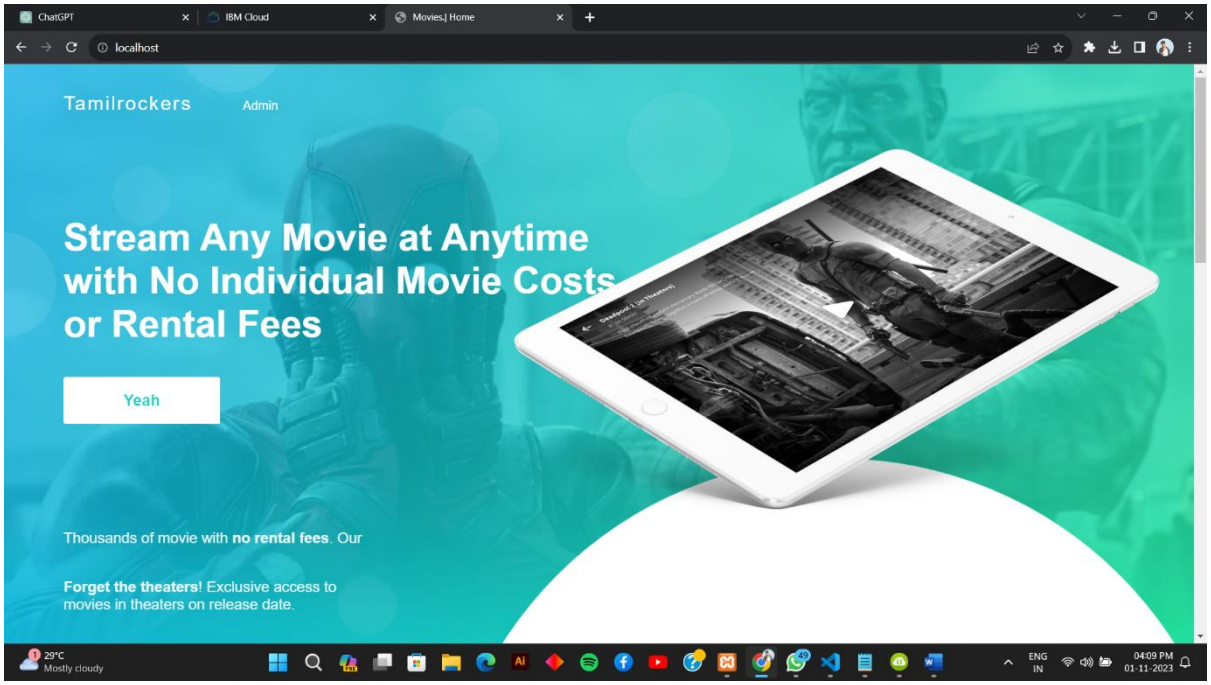
- If the connection fails, it terminates the script with an error message using `die()`. This is usually not recommended for production code; proper exception handling should be used instead.

- If the connection is successful, it assigns the created connection to the static property `$conn` and returns the connection.

5. Connection Reuse:

- If a connection already exists (`$conn` is not `null`), the method simply returns the existing connection. This ensures that only one database connection is created and reused throughout the application, promoting efficiency.

Screenshot :



Conclusion :

In summary, this code defines a `Database` class with a static method that manages a single database connection. It checks if a connection exists, creates one if it doesn't, and then returns the connection. If a connection already exists, it reuses the existing one. This class is designed to encapsulate database connection logic and promote code reusability.

In summary, this code combines PHP and HTML to create a web page. It first establishes a database connection, retrieves movie data from a database, and then dynamically generates an HTML page to display a list of movies with associated images, links, and other elements. It includes a header, a movie list, a call to action, and a footer for a basic web page structure.