Day 5: Working with Superglobals + Dashboard Project (With Bootstrap)

Slide 1: Introduction to Superglobals in PHP

⏱️Time: 5 minutes

Explanation: Superglobals are built-in variables in PHP that are always accessible, regardless of scope. These include:

- `$_GET`
- `$_POST`
- `$_REQUEST`
- `$_SERVER`
- `$_SESSION`
- `$_COOKIE`
- `$_FILES`

We will explore how to use them securely and effectively in a Bootstrap-styled environment.

Task: Print out your current page's URL, browser, and IP address using `$_SERVER`. Style it with Bootstrap cards.

Answer:

```
<div class="container mt-4">
  <div class="row">
    <div class="col-md-4">
      <div class="card"><div class="card-body">
        <strong>URL:</strong> <?= $_SERVER['REQUEST_URI'] ?>
      </div></div>
    </div>
    <div class="col-md-4">
      <div class="card"><div class="card-body">
        <strong>Browser:</strong> <?= $_SERVER['HTTP_USER_AGENT'] ?>
      </div></div>
    </div>
    <div class="col-md-4">
      <div class="card"><div class="card-body">
        <strong>IP:</strong> <?= $_SERVER['REMOTE_ADDR'] ?>
      </div></div>
    </div>
  </div>
</div>
```

Slide 2: Validating Access Using `$_SERVER`

⏱️Time: 5 minutes

Explanation: Use `$_SERVER` to ensure the user is accessing from a specific host, or to block certain IPs.

Task: Block access to your page if user is not visiting from `localhost`. Show Bootstrap error alert.

Answer:

```php
<?php
if ($_SERVER['HTTP_HOST'] !== 'localhost') {
  echo '<div class="alert alert-danger m-3">Access denied: Invalid host.</div>';
  exit;
}
?>
```

Slide 3: Session Introduction and User Counter

⏱️Time: 5 minutes

Explanation: Sessions are used to store data across multiple page loads.

Task: Increment a session variable each time user refreshes and show the count inside a Bootstrap badge.

Answer:

```php
<?php
session_start();
$_SESSION['visits'] = ($_SESSION['visits'] ?? 0) + 1;
?>
<p class="m-3">You visited this page <span class="badge bg-primary"><?=
$_SESSION['visits'] ?></span> times.</p>
```

Slide 4: Add User to Session

⏱️Time: 5 minutes

Explanation: We can store arrays (like user data) in session. This helps in dashboard login or data accumulation.

Task: Create a Bootstrap form to enter name and email, and store the user in session.

Answer:

```
<?php
session_start();
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
  $_SESSION['users'][] = [
    'name' => $_POST['name'],
    'email' => $_POST['email']
  ];
}
?>
<form method="post" class="p-3">
  <input type="text" name="name" placeholder="Name" class="form-control mb-2"
required>
  <input type="email" name="email" placeholder="Email" class="form-control mb-2"
required>
  <button type="submit" class="btn btn-success">Save</button>
</form>
```

Slide 5: List Users from Session in Table

⏱️Time: 5 minutes

Task: Loop over saved users and display them in a Bootstrap table.

Answer:

```
<?php
$users = $_SESSION['users'] ?? [];
if ($users): ?>
<table class="table table-striped m-3">
  <thead><tr><th>#</th><th>Name</th><th>Email</th></tr></thead>
  <tbody>
    <?php foreach ($users as $i => $user): ?>
      <tr><td><?= $i+1 ?></td><td><?= $user['name'] ?></td><td><?=
$user['email'] ?></td></tr>
    <?php endforeach; ?>
  </tbody>
</table>
<?php endif; ?>
```

Slide 6: Bootstrap Modal & Error Validation

⏱️Time: 7 minutes

Task: Create a form with Bootstrap validation, and show a modal when the data is successfully stored.

Answer: (HTML Only Snippet)

```
<!-- Form with Validation -->
<form method="post" class="needs-validation" novalidate>
  <input name="name" class="form-control mb-2" required>
  <input name="email" type="email" class="form-control mb-2" required>
  <button class="btn btn-primary">Submit</button>
</form>

<!-- Modal -->
<div class="modal fade" id="successModal">
  <div class="modal-dialog"><div class="modal-content">
    <div class="modal-body">User added successfully!</div>
  </div></div>
</div>
```

(Use JS to trigger modal after successful submit)

Slide 7: Security & IP Filtering

⏱️Time: 6 minutes

Task: Use `$_SERVER['REMOTE_ADDR']` to filter/block a range of IPs and redirect if blocked.

Answer:

```
$blocked_ips = ['192.168.0.1'];
if (in_array($_SERVER['REMOTE_ADDR'], $blocked_ips)) {
  header("Location: blocked.php");
  exit;
}
```

Slide 8: Store JSON of Users in Cookie

⏱️Time: 6 minutes

Explanation: Storing multiple users in cookies as JSON string.

Task: On user submission, save array in cookie for 1 hour.

Answer:

```
$users = json_decode($_COOKIE['users'] ?? '[]', true);
$users[] = ['name' => $_POST['name'], 'email' => $_POST['email']];
setcookie('users', json_encode($users), time() + 3600);
```

Slide 9: Secure User Authentication using Native PHP Hashing

⏱️Time: 7 minutes

Explanation: Instead of storing plain passwords, use `password_hash()` and `password_verify()` for safe authentication.

Task: Create user registration and login using secure password hashing.

Register:

```
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
  $name = $_POST['name'];
  $email = $_POST['email'];
  $password = password_hash($_POST['password'], PASSWORD_DEFAULT);

  $_SESSION['users'][] = ['name' => $name, 'email' => $email, 'password' =>
$password];
}
```

Login:

```
foreach ($_SESSION['users'] as $user) {
  if ($user['email'] == $_POST['email'] && password_verify($_POST['password'],
$user['password'])) {
    $_SESSION['auth'] = $user;
    echo "<div class='alert alert-success'>Login successful. Welcome
{$user['name']}!</div>";
    exit;
  }
```

```
}
echo "<div class='alert alert-danger'>Invalid credentials.</div>";
```

Final Task: Mini Dashboard Project

⏱️Time: 35–40 minutes

Features:

- Add User Form with Bootstrap Validation.
- Display Users in Table.
- Modal after Submission.
- IP Filtering using `$_SERVER`.
- Session Count.
- Secure Login using `password_hash()`.
- Optional: Store JSON in Cookie.

Output: Final mini dashboard showing real usage of all superglobals styled in Bootstrap.