Protecting Pages and Preventing Unauthorised Access

A protected page checks authentication on the server before sending restricted content. Hiding a menu link is not protection because a user can enter the URL directly.

Require a logged-in user

Place this before any HTML:

```
<?php
session_start();

if (!isset($_SESSION["user_id"])) {
    header("Location: login.php");
    exit;
}
```

The `exit` prevents the rest of the page from running after the redirect.

Complete protected page

```
<?php
session_start();

if (!isset($_SESSION["user_id"])) {
    header("Location: login.php");
    exit;
```

```
}

$username = $_SESSION["username"] ?? "user";
?>
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Dashboard</title>
</head>
<body>
  <h1>User dashboard</h1>
  <p>Welcome, <?= htmlspecialchars($username) ?>.</p>
</body>
</html>
```

Use a reusable guard

Create `includes/require-login.php`:

```
<?php
if (session_status() !== PHP_SESSION_ACTIVE) {
    session_start();
}

if (!isset($_SESSION["user_id"])) {
    header("Location: login.php");
    exit;
}
```

Then begin protected pages with:

```
require __DIR__ . "/includes/require-login.php";
```

Test direct access

Use a private browser window or log out, then enter the protected URL directly. Also test after closing the browser and after destroying the session.

Check

- [ ] Authentication is checked before output.
- [ ] Redirect is followed by `exit`.
- [ ] Every protected processing route uses the guard.
- [ ] Direct URL access is tested.
- [ ] Displayed session values are escaped.

Revision #2
Created 2026-06-08 08:45:27 UTC by Mr Napper
Updated 2026-08-18 23:49:50 UTC by Mr Napper