# GWS Portable Config & Deployment Guide (Sprint 0 → Production)

Portable setup for XAMPP (local) and cPanel (production), with bootstrap detection, environment-aware config, .htaccess examples, cPanel env variables, and migration/parity checklists.

## Assumptions

• PROJECT\_ROOT contains /private and /public\_html

• public\_html is web root; /private holds non-public PHP

• May deploy at domain root or under /public\_html/subfolder/

• PHP 8.x; PDO available; mysqli fallback

## Files to Add/Update

• private/bootstrap.php — discovers PROJECT\_ROOT; defines BASE\_PATH/BASE\_URL

• private/gws-universal-config.php — env-aware, no hard-coded paths; HTTPS-aware sessions

• .htaccess — pretty URLs (root or subfolder)

• cPanel env vars — SetEnv for DB creds and APP\_ENV

## private/bootstrap.php (drop-in)

<?php

declare(strict\_types=1);

/\*\* Find PROJECT\_ROOT by walking up until /private exists \*/

function find\_project\_root(string $startDir): string {

$dir = $startDir;

while ($dir !== dirname($dir)) {

if (is\_dir($dir . DIRECTORY\_SEPARATOR . 'private')) return $dir;

$dir = dirname($dir);

}

throw new RuntimeException('Project root not found.');

}

define('PROJECT\_ROOT', find\_project\_root(\_\_DIR\_\_));

define('PRIVATE\_ROOT', PROJECT\_ROOT . '/private');

define('PUBLIC\_ROOT', PROJECT\_ROOT . '/public\_html');

/\*\* Detect base path ("/" or "/subfolder") and base URL \*/

function detect\_base\_path(): string {

if (PHP\_SAPI === 'cli') return '/';

$docRoot = rtrim(str\_replace('\\','/', $\_SERVER['DOCUMENT\_ROOT'] ?? ''), '/');

$public = rtrim(str\_replace('\\','/', realpath(PUBLIC\_ROOT) ?: ''), '/');

if ($docRoot && $public && str\_starts\_with($public, $docRoot)) {

$base = substr($public, strlen($docRoot));

return $base === '' ? '/' : (str\_starts\_with($base, '/') ? $base : '/'.$base);

}

$script = $\_SERVER['SCRIPT\_NAME'] ?? '/';

$pos = strrpos($script, '/');

return $pos === false ? '/' : substr($script, 0, $pos) ?: '/';

}

function detect\_base\_url(): string {

if (PHP\_SAPI === 'cli') return 'http://localhost';

$https = (!empty($\_SERVER['HTTPS']) && $\_SERVER['HTTPS'] !== 'off')

|| (($\_SERVER['HTTP\_X\_FORWARDED\_PROTO'] ?? '') === 'https');

$scheme = $https ? 'https' : 'http';

$host = $\_SERVER['HTTP\_X\_FORWARDED\_HOST']

?? $\_SERVER['HTTP\_HOST']

?? $\_SERVER['SERVER\_NAME']

?? 'localhost';

$port = $\_SERVER['HTTP\_X\_FORWARDED\_PORT'] ?? $\_SERVER['SERVER\_PORT'] ?? null;

$portPart = ($port && !in\_array((int)$port, [80,443], true) && !str\_contains($host, ':')) ? ':' . $port : '';

$basePath = detect\_base\_path();

$url = rtrim("$scheme://$host$portPart" . rtrim($basePath, '/'), '/');

return $url ?: "$scheme://$host$portPart";

}

define('BASE\_PATH', detect\_base\_path()); // "/" or "/subfolder"

define('BASE\_URL', detect\_base\_url()); // "https://site.tld" or "https://site.tld/subfolder"

define('ASSETS\_URL', BASE\_URL . '/assets'); // convenience

// Load your universal config + helpers

require\_once PRIVATE\_ROOT . '/gws-universal-config.php';

require\_once PRIVATE\_ROOT . '/gws-universal-functions.php';

// (Optional) Composer autoload at project root

if (is\_file(PROJECT\_ROOT . '/vendor/autoload.php')) {

require\_once PROJECT\_ROOT . '/vendor/autoload.php';

}

## private/gws-universal-config.php (refactor)

<?php

declare(strict\_types=1);

// Timezone

date\_default\_timezone\_set('America/New\_York');

// Environment

define('APP\_ENV', getenv('APP\_ENV') ?: 'dev'); // dev|staging|production

// Session (HTTPS-aware)

if (session\_status() === PHP\_SESSION\_NONE) {

$https = (!empty($\_SERVER['HTTPS']) && $\_SERVER['HTTPS'] !== 'off')

|| (($\_SERVER['HTTP\_X\_FORWARDED\_PROTO'] ?? '') === 'https');

ini\_set('session.use\_only\_cookies', '1');

ini\_set('session.cookie\_httponly', '1');

ini\_set('session.cookie\_secure', $https ? '1' : '0');

ini\_set('session.cookie\_samesite', 'Lax');

ini\_set('session.gc\_maxlifetime', '3600');

ini\_set('session.cookie\_lifetime', '0');

ini\_set('session.gc\_probability', '1');

ini\_set('session.gc\_divisor', '100');

session\_start();

if (!isset($\_SESSION['CREATED'])) {

$\_SESSION['CREATED'] = time();

} elseif (time() - $\_SESSION['CREATED'] > 3600) {

session\_regenerate\_id(true);

$\_SESSION = [];

$\_SESSION['CREATED'] = time();

}

}

// Paths (defaults if bootstrap not loaded yet)

if (!defined('PROJECT\_ROOT')) define('PROJECT\_ROOT', dirname(\_\_DIR\_\_));

if (!defined('PRIVATE\_ROOT')) define('PRIVATE\_ROOT', PROJECT\_ROOT . '/private');

if (!defined('PUBLIC\_ROOT')) define('PUBLIC\_ROOT', PROJECT\_ROOT . '/public\_html');

if (!defined('BASE\_PATH')) define('BASE\_PATH', '/');

if (!defined('BASE\_URL')) define('BASE\_URL', 'http://localhost');

if (!defined('ASSETS\_URL')) define('ASSETS\_URL', BASE\_URL . '/assets');

// Env helper

function env(string $k, $d=null){ $v=getenv($k); return $v!==false?$v:$d; }

// Database (env-driven with dev defaults)

define('DB\_HOST', env('DB\_HOST','localhost'));

define('DB\_NAME', env('DB\_NAME','gws\_universal\_db'));

define('DB\_USER', env('DB\_USER','root'));

define('DB\_PASS', env('DB\_PASS',''));

define('DB\_CHARSET', env('DB\_CHARSET','utf8mb4'));

define('SECRET\_KEY', env('SECRET\_KEY','change-me-please'));

// Back-compat lowercase constants

if (!defined('db\_host')) define('db\_host', DB\_HOST);

if (!defined('db\_name')) define('db\_name', DB\_NAME);

if (!defined('db\_user')) define('db\_user', DB\_USER);

if (!defined('db\_pass')) define('db\_pass', DB\_PASS);

if (!defined('db\_charset')) define('db\_charset', DB\_CHARSET);

if (!defined('secret\_key')) define('secret\_key', SECRET\_KEY);

// Public URLs (no /public\_html in URLs)

if (!defined('BLOG\_ASSETS\_URL')) define('BLOG\_ASSETS\_URL', BASE\_URL . '/blog\_system/assets');

if (!defined('PUBLIC\_ASSETS\_URL')) define('PUBLIC\_ASSETS\_URL', BASE\_URL . '/assets');

if (!defined('BACKGROUND\_IMAGES\_URL')) define('BACKGROUND\_IMAGES\_URL',BASE\_URL . '/blog\_system/assets/settings/background\_image');

if (!defined('ACCOUNTS\_AVATARS\_URL')) define('ACCOUNTS\_AVATARS\_URL', BASE\_URL . '/accounts\_system/assets/uploads/avatars');

if (!defined('BLOG\_AVATARS\_URL')) define('BLOG\_AVATARS\_URL', BASE\_URL . '/blog\_system/assets/uploads/img/avatars');

if (!defined('BLOG\_POST\_IMAGES\_URL')) define('BLOG\_POST\_IMAGES\_URL', BASE\_URL . '/blog\_system/assets/uploads/img/blog\_post\_images');

if (!defined('BLOG\_GALLERY\_URL')) define('BLOG\_GALLERY\_URL', BASE\_URL . '/blog\_system/assets/uploads/img/gallery');

// Filesystem paths (portable)

if (!defined('public\_path')) define('public\_path', PUBLIC\_ROOT);

if (!defined('private\_path')) define('private\_path', PRIVATE\_ROOT);

if (!defined('admin\_path')) define('admin\_path', PUBLIC\_ROOT . '/admin');

if (!defined('documents\_system\_path')) define('documents\_system\_path', PUBLIC\_ROOT . '/documents\_system');

if (!defined('vendor\_path')) define('vendor\_path', PROJECT\_ROOT . '/vendor');

if (!defined('public\_assets\_path')) define('public\_assets\_path', PUBLIC\_ROOT . '/assets');

if (!defined('blog\_path')) define('blog\_path', PUBLIC\_ROOT . '/blog\_system');

if (!defined('client\_portal\_path')) define('client\_portal\_path', PUBLIC\_ROOT . '/client\_portal');

if (!defined('accounts\_system\_path')) define('accounts\_system\_path', PUBLIC\_ROOT . '/accounts\_system');

// Branding (fallbacks)

if (!defined('BUSINESS\_NAME')) define('BUSINESS\_NAME', 'Your Business Name');

if (!defined('BUSINESS\_SHORT\_NAME')) define('BUSINESS\_SHORT\_NAME', 'YBN');

if (!defined('ADMIN\_LOGO')) define('ADMIN\_LOGO', 'admin\_logo.svg');

if (!defined('MAIN\_LOGO')) define('MAIN\_LOGO', 'main\_logo.png');

if (!defined('FAVICON')) define('FAVICON', 'favicon.ico');

if (!defined('SECONDARY\_LOGO')) define('SECONDARY\_LOGO', 'secondary\_logo.png');

if (!defined('BRANDING\_ASSETS\_URL')) define('BRANDING\_ASSETS\_URL', BASE\_URL . '/assets/branding');

if (!defined('BRANDING\_ASSETS\_PATH'))define('BRANDING\_ASSETS\_PATH', public\_assets\_path . '/branding');

if (!defined('BRAND\_PRIMARY\_COLOR')) define('BRAND\_PRIMARY\_COLOR', '#667eea');

if (!defined('BRAND\_SECONDARY\_COLOR')) define('BRAND\_SECONDARY\_COLOR', '#764ba2');

if (!defined('BRAND\_ACCENT\_COLOR')) define('BRAND\_ACCENT\_COLOR', '#28a745');

if (!defined('BRAND\_GRADIENT')) define('BRAND\_GRADIENT', 'linear-gradient(135deg, ' . BRAND\_PRIMARY\_COLOR . ' 0%, ' . BRAND\_SECONDARY\_COLOR . ' 100%)');

if (!defined('BRAND\_FONT\_FAMILY')) define('BRAND\_FONT\_FAMILY', 'system-ui, -apple-system, "Segoe UI", Roboto, sans-serif');

if (!defined('BRAND\_HEADING\_FONT')) define('BRAND\_HEADING\_FONT', BRAND\_FONT\_FAMILY);

// Composer autoload

$autoloads = [PROJECT\_ROOT . '/vendor/autoload.php', documents\_system\_path . '/vendor/autoload.php'];

foreach ($autoloads as $a) { if (is\_file($a)) { require\_once $a; break; } }

// Include helpers

require\_once PRIVATE\_ROOT . '/gws-universal-functions.php';

// Database connections

$pdo = null; $mysqli = null;

if (class\_exists('PDO')) {

try {

$pdo = new PDO("mysql:host=".DB\_HOST.";dbname=".DB\_NAME.";charset=".DB\_CHARSET, DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION,

PDO::ATTR\_DEFAULT\_FETCH\_MODE => PDO::FETCH\_ASSOC,

]);

} catch (Throwable $e) {

if (APP\_ENV !== 'dev') error\_log('PDO connection failed: ' . $e->getMessage());

$pdo = null;

}

}

if (!$pdo && class\_exists('mysqli')) {

try {

$mysqli = @new mysqli(DB\_HOST, DB\_USER, DB\_PASS, DB\_NAME);

if ($mysqli->connect\_error) throw new RuntimeException($mysqli->connect\_error);

$mysqli->set\_charset(DB\_CHARSET);

} catch (Throwable $e) {

if (APP\_ENV !== 'dev') { error\_log('MySQLi connection failed: ' . $e->getMessage()); die('A system error occurred. Please try again later.'); }

else { die('MySQLi Database connection failed: ' . $e->getMessage()); }

}

}

if (!$pdo && !$mysqli) {

if (APP\_ENV !== 'dev') { error\_log('No DB extension available (PDO/mysqli)'); die('A system error occurred. Please try again later.'); }

else { die('Database connection failed: Neither PDO nor MySQLi extensions are available.'); }

}

// DB helpers

function db\_query($sql, $params = []) {

global $pdo, $mysqli;

if ($pdo) { $stmt=$pdo->prepare($sql); $stmt->execute($params); return $stmt; }

if ($mysqli) {

if (empty($params)) { $res=$mysqli->query($sql); if(!$res) throw new Exception('MySQLi query failed: '.$mysqli->error); return $res; }

$stmt=$mysqli->prepare($sql); if(!$stmt) throw new Exception('MySQLi prepare failed: '.$mysqli->error);

$types=str\_repeat('s', count($params)); $stmt->bind\_param($types, ...$params); $stmt->execute(); return $stmt;

}

throw new Exception('No database connection available');

}

function db\_fetch\_all($result) {

global $pdo, $mysqli;

if ($pdo && $result instanceof PDOStatement) return $result->fetchAll();

if ($mysqli && $result instanceof mysqli\_result) return $result->fetch\_all(MYSQLI\_ASSOC);

if ($mysqli && $result instanceof mysqli\_stmt) { $rs=$result->get\_result(); return $rs ? $rs->fetch\_all(MYSQLI\_ASSOC) : []; }

return [];

}

## .htaccess Examples

Root deployment (public\_html):

RewriteEngine On

RewriteBase /

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule ^ index.php [L]

Subfolder deployment (public\_html/app):

RewriteEngine On

RewriteBase /app/

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule ^ index.php [L]

## cPanel Environment Variables (.htaccess)

SetEnv APP\_ENV production

SetEnv DB\_HOST localhost

SetEnv DB\_NAME USER\_dbname

SetEnv DB\_USER USER\_dbuser

SetEnv DB\_PASS "super-secret"

## Migration Checklist (Old Laptop → New Laptop)

☐ Commit & push all changes (git add -A; git commit; git push)

☐ Export DBs as SQL dumps via phpMyAdmin (do NOT copy mysql/data)

☐ Copy XAMPP configs if customized (httpd.conf, vhosts.conf, php.ini)

☐ Export VS Code extensions (code --list-extensions > extensions.txt) and settings.json

☐ Zip non-git assets (uploads/, storage/, images/)

☐ Record hosts file entries (if using \*.local domains)

## New Laptop Setup

☐ Install XAMPP (matching major PHP version)

☐ Install VS Code and Git; import extensions.txt

☐ Clone repo into htdocs (or copy with .git intact)

☐ Create databases and Import SQL dumps

☐ Restore php.ini and vhosts if needed; restart Apache/MySQL

☐ composer install (if applicable)

☐ Add hosts entries for local domains if used

☐ Open project in VS Code (workspace with path "." is portable)

## 10-Minute Parity Test (Local → cPanel)

☐ Load a page that includes CSS/JS/images (no /public\_html in URLs)

☐ Log in and perform one write action (e.g., form submit)

☐ Upload a file and confirm it appears in the right folder

☐ If in subfolder, verify links respect BASE\_PATH and .htaccess RewriteBase

☐ Verify PHP extensions on cPanel via phpinfo() (PDO, mysqli, gd/imagick, curl, openssl)

## Day-to-Day Git Tips

• git pull --rebase; git push (cleaner history)

• Use --force-with-lease only when replacing remote history intentionally

• Add .gitattributes with '\* text=auto' to normalize line endings