Goal

To create a manageable and secure process for keeping your PHP version up to date, specifically one release behind the major release each December. If there are major problems with the upgrade contact the client to determine a path forward.

Establish current state and plan resources. (TBD once a year)

Inventory All Sites

Note PHP versions, CMS/framework used, dependencies, and hosting environments.

Define Target Version (what we are going to upgrade too, one release behind normally)

Phase 1: Upgrade the Outdated Sites

Once a year make sure all sites are updated.

Prioritize by Risk

Tackle less complex sites first

Reserve more complex or mission-critical sites for last

See Upgrade & Rollback Workflow for PHP Version Testing per site for process for each site

Upgrade & Rollback Workflow for PHP Version Testing per site

Approach: Upgrade live site during off-hours → test → rollback if issues → set up dedicated test environment

Pre-Upgrade Preparation (Before Off-Hours Window)

- Full Site Backup
- Version Compatibility Check
- Flag known incompatible functions/features
- Note Current PHP Version
- Log it (important for rollback)
- Prepare Rollback Script or Snapshot

Off-Hours Upgrade Process

- Timing: Schedule a 1–2 hour window during low traffic
- Switch PHP Version
- Update server config to target version (e.g., from PHP 7.4 to PHP 8.2)
- Restart services (Apache/Nginx + PHP-FPM)

- Verify Upgrade is Applied
- Run php -v or phpinfo() page to confirm
- Testing (Manual/Automated)
- Load key pages: homepage, forms, admin panel
- Check logs: error.log, php-fpm.log
- If Site Breaks → Rollback Immediately
- Revert to Previous PHP Version
- Reset server config to earlier PHP version
- Restart services
- Restore from Backup if Needed
- Log Errors Encountered
- Copy logs and note problematic pages/

If there is an issue not easily fixable contact client to determine a path forward once you get approval

Setup Dedicated Test Environment

- Clone the site (files + DB) into a staging server
- Install and configure the target PHP version
- Reproduce the issue
- Begin fixing compatibility issues iteratively
- Once Fixed in Staging → Try Upgrade Again
- Re-apply upgrade during off-hours