



Document	Product Operation Guide	Release	17/08/26 V1.0
Project	BRN WMS	Project code	200-WMS-26-001-00
Project period	05/01/26–24/08/26		

Objective

Guide a System Administrator through installing, configuring, operating, monitoring, backing up, and recovering BRN WMS, so operation is consistent, correct, and quickly recoverable.

1. Deployment options

BRN WMS supports two deployment paths, evidenced in the repository:

Path	Mechanism	Reference
Manual installation	One-shot database setup script	<code>setup.php</code>
Containerized deployment	Docker Compose stack: php-apache, mariadb, node/pm2	<code>docker-compose.yml</code> , <code>docker/</code>

2. Containerized installation (recommended)

1. Copy `.env.example` to `.env`, or run the interactive generator: `docker/init-env.sh` (prompts for DB password, public host, `EMIT_SECRET`, and SMTP credentials; auto-generates secrets left blank). 2. Run `docker compose up -d --build`. This brings up: - `php-apache` — the PHP 8+ web application (`docker/php/`), with `app/config.php` generated from `.env` at container start by `docker/php/entrypoint.sh` — never baked into the image or committed. - `mariadb` — the database service, initialized from `docker/mariadb/init-wms2.sql` and `setup.php`. - `node` — the Node.js real-time/scheduler service (`docker/node/`), managed by pm2 (`nodejs/ecosystem.config.js`). 3. Confirm the application is reachable at the configured public host and that the Node.js service is running (Section 5).

3. Manual installation

1. Provision a PHP 8+ / MariaDB / Node.js environment. 2. Configure `app/config.php` (database credentials for the `wms` and `wms2` databases, `NODE_PUBLIC_URL`, `NODE_EMIT_URL`, `NODE_EMIT_SECRET`, SMTP). 3. Run `setup.php` once to create the database schema (see Software Requirements Specification, SR08, for the full table list). 4. Start the Node.js services (`nodejs/server.js`, `nodejs/scheduler.js`), for example under pm2 using `nodejs/ecosystem.config.js`.

4. Configuration

Item	Location	Notes
Application configuration	<code>app/config.php</code>	Generated at deploy time; never committed
Environment secrets	<code>.env</code> (Docker) or shell/deployment environment (manual)	Excluded via <code>.gitignore</code>
Company-level settings	In-application (<code>app/setting/</code>)	Per-company profile, SMTP, system settings
Time zone	<code>\$time_zone</code> in <code>app/config.php</code>	Fixed to Asia/Bangkok

5. Monitoring

- Web application: confirm the PHP application responds and users can authenticate.
- Database: confirm both `wms` and `wms2` databases are reachable.
- Node.js service: confirm `server.js` (Socket.IO) and `scheduler.js` (scheduled jobs) are running under pm2; review `nodejs/logs/` (`socket.log`, `scheduler.log`, `scheduler-error.log`) for errors.
- Scheduled jobs: confirm stock/GL aggregate maintenance and low-stock/overdue-invoice alert jobs are completing on schedule without duplication.

6. Backup and recovery

Item	Mechanism	Status
Source code and configuration templates	Git, two remotes (<code>origin</code> , <code>backup</code>)	See Project Repository / Project Repository (Backup), work products 9–10

Item	Mechanism	Status
Database (wms, wms2)	Scheduled <code>mysqldump</code> script, run daily, output stored off-server with a retention policy	Reported by the Developer (17/08/26); script location, exact schedule, off-server destination, and retention period are not yet recorded in a controlled configuration reference, and no restoration has been tested — see Section 7
SDLC documents	External PDF export package	See Project Repository (Backup), Section 3

7. Outstanding operational gaps

ID	Gap	Owner	Required before
OP-001	A daily <code>mysqldump</code> backup exists for wms/wms2 (developer-reported), but its script location, exact schedule, off-server destination, and retention period are not yet recorded in a controlled reference, and no restoration has been tested.	Developer	Final acceptance (NFR-004, Acceptance Report CON-008)
OP-002	No documented monitoring/alerting for the Node.js service beyond log files.	Developer	Operational acceptance

8. User and access management

An Owner/Admin manages users, roles (Owner/Admin/Staff/Viewer), and application-access flags from Settings (`app/setting/`). Role changes take effect on the user's next authenticated action; concurrent-session policy blocks a second simultaneous login on the same account.

Approval

Prepared by

Name: Thanakorn Sathitwitayakul

Role: Developer

Signature: _____

Date: _____

Reviewed by

Name: Apirach Supattaratpateep

Role: Project Manager

Signature: _____

Date: _____

Reviewed and authorized by

Name: Seri Viriyasakultorn

Project roles: Project Sponsor / Customer Representative / Authorized Approver

Position: Managing Director

Company: B.R.N. Enterprise Co., Ltd.

Signature: _____

Date: _____