

# Vulper

[<https://github.com/GSCloud/vulper>]

Base PHP 8.1 docker container app.

Deploys basic containers **app**, **db**, **cache** (*Redis*) and **phpmyadmin** (*optional*).

□ Images, extensions, external binaries, ports and volumes are freely configurable.

## Prerequisites

- Docker environment [<https://docs.docker.com/get-docker/>]
- `sudo apt install make` - utility to process Makefile
- `sudo apt install ruby-full` - Rusy is used for YAML 2 JSON conversions only

## Makefile

Run `make` for help.

## Application Logic

### Web

- `www/index.php` (Apache root directory is `www/` by default)
- `app/Bootstrap.php`
- `app/Web.php`
- custom controllers (MVC pattern)

### CLI or CRON

- `cli.sh` (shortcut for loading Bootstrap)
- `app/Bootstrap.php`
- `app/Cli.php`
- custom controllers (MVC pattern)

### Bootstrap

**Bootstrap.php** performs these tasks: - set constants - import Composer autoloader (run `make install` or `make update`) - load config and router definitions - setup debugger - execute the Web or Cli controller

□ Debugger can be extensively configured via **config.neon**.

## Web Controller

**Web.php** performs these tasks: - read routing table configuration - find the corresponding controller - continue the workflow - add extra headers

## Cli Controller

**Cli.php** performs these tasks: - check command line parameters - find the corresponding controller - continue the workflow / display a help message to TTY

## Configuration

Configuration is stored as **.env** file. Demo configurations are available as **env...-redis** files.

□ Copy a configuration file over the **.env** and run **make**.

- **.env** - environment configuration
- **app/env.json** - generated file = .env translated to JSON format (□ run **make savejson** after the configuration changes)
- **app/config.neon** - main app config [<https://doc.nette.org/en/neon>]
- **app/router.neon** - main app router
- **www/.htaccess** - Apache 2 htaccess
- **INI** / *Docker injected configuration files* to alter resource limits, post and upload limits, default timezone
- **CLI\_REQ** - environment variable to use if CLI SAPI is detected = set it to specify a PHP required file that can load extra data, alter settings or override defaults

□ Make sure to remove containers before changing ports in the **.env**.

□ There is also an **env.mustache** template available for programmatic environment creation.

## Constants

- **DS** - directory separator shortcut
- **CLI** - true if CLI SAPI is detected
- **LOCALHOST** - true if localhost is detected
- **ROOT** - full path to the Vulper root
- **APP** - full path to app/ folder
- **WWW** - full path to www/ folder

## Configuration Commands

- **make cfg** - show docker-compose configuration
- **make jsoncfg** - show docker-compose configuration as JSON
- **make jsonapp** - show environment as JSON

- `make savejson` - save application config to app/env.json as JSON

## Router

Web routes are stored in **router.neon**.

Array *defaults* is applied to all the routes, every route must set at least: *path*, *controller* and *view*, if not using the default values.

## Installation

- `make install` - create docker images
- `make install extensions` - create docker images, add PHP extensions afterwards
- `make check` - check running containers

## Updates

- `make update` - update installation

## Development

- `make exec` - run Bash in the app container
- `make applog` - show app container logs
- `make csfixer` - run PHP CS-FIXER in app/
- `make phpstan` - run PHPStan static analysis in app/
- `make test` - run Nette tester tests in app/

## Container Operations

- `make stop` - stop containers
- `make start` - resume stopped containers

Always use `install` to create containers if they got removed.

□ phpMyAdmin (PMA) container can be disabled via `.env` by setting `PMA_DISABLE=1`

## Cleaning and Removal

- `make remove` - remove containers
- `make purge` - remove containers + database folder

## Logging

Tracy logs and exceptions are mapped outside the container and available at:

`/tmp/${APP_NAME}/logs`

## To Do

- Dockerfile
  - build
  - push
- database access
- database operations
  - export
  - import
- tests
  - codeception