Plunner

Plunner

Installation and setup guide

Table of Contents

1. Installation

Plunner consists in a frontend and in a backend part, ideally these parts should be installed into different hardware tiers but it is also possible to use single hardware tier with two different virtual hosts

1.1 Github repositories

- Frontend: https://github.com/dsd-meetme/frontend
- Backend: https://github.com/dsd-meetme/backend

1.2 Companion files of this document

As a companion of this document a set of two zip files is included: these zips contain all the files of the frontend and of the backend of the project and so can be used to install the project instead of using the github repos listed above.

1.1 Prerequisites

- NodeJS (https://nodejs.org/) [for frontend]
- Composer (https://getcomposer.org/) [for backend]
- GLPSOL (https://en.wikibooks.org/wiki/GLPK/Linux_packages) [for backend]
- Apache http server version 2.* (https://httpd.apache.org/) [for backend]

1.2 Frontend installation

Here are presented two different ways to install the frontend part of Plunner.
Note that if u want to access Plunner remotely you have to install it in the public folder of your virtual host

1.2.1 Classical installation

1. Clone the frontend github repository on your local machine into a folder `F` or extract the frontend zip archive included with this document into a folder `F`
2. Open your terminal, move into the subfolder `frontend` of the folder `F` and type `npm install` to install all the dependencies required by this part of the project (if you want to install only the production dependencies of this part of the project type instead `npm install --production`)

3. Still in your terminal, type `node_modules/.bin/gulp production` to bootstrap the application you can access to it from your browser by opening the index.html file in `frontend` subfolder of the folder `F`

1.2.2 Simplified installation

1. Open your terminal, and install the create-project utility by typing `npm install -g create-project`
2. Still in your terminal, move into a folder and type `create-project frontend dsd-meetme/frontend`, your should see a new folder called “frontend” with all the files of the frontend of Plunner
3. Still in your terminal, move into the “frontend” folder and type `node_modules/.bin/gulp production` to bootstrap the application you can access to it from your browser by opening the index.html file in “frontend” folder

1.3 Backend installation

Here are presented two different ways to install the frontend part of Plunner.
Note that if u want to access Plunner remotely you have to install it in the public folder of your virtual host

1.3.1 Classical installation

1. Clone the backend github repository on your local machine into a folder `F` or extract the backend.zip archive included with this document into a folder `F`
2. Open your terminal, move into the subfolder `backend` of the folder `F` and type `composer install` to install all the dependencies required by this part of the project

1.3.2 Simplified installation

1. Open your terminal and move to a folder `F`
2. Still in your terminal, type `composer create-project dsd-meetme/backend`, this will install and bootstrap the backend of Plunner in the backend subfolder of the folder `F`

2. Configuration

Here follows a series of instructions to how to configure the backend and the frontend of Plunner

2.1 Frontend configuration

You can configure the url the frontend will use to communicate to the backend of Plunner by editing the config.js file in the project's folder as described in the file itself

2.2 Backend configuration

We are in the context of the backend project's folder

1. Create a MySql database
2. Configure database credentials in .env file
3. Generate a JWT_SECRET key by typing `php artisan jwt:generate,` paste this key in the corresponding field in the .env file
4. Generate an APP_KEY key by typing `php artisan key:generate`, paste this key in the corresponding field in the .env file
5. Type `php artisan migrate`
6. Configure urls in config/app.php (this only for real environment)
7. Configure Plunner's periodic tasks using unix crontab, by typing `crontab * * * * * php /path/to/artisan schedule:run >> /dev/null 2>&1`
8. Configure additional things like emails, optimisation and so in the config files

3. For developers

3.1 Frontend development mode

After the installation of the frontend of Plunner, a development mode can be use to facilitate and smooth the testing and the development process of the application:

- Type `gulp dev` to bootstrap a local webserver that runs Plunner’s frontend on port 3000
- Type `karma start` to bootstrap a local webserver that runs Plunner’s frontend unit tests on port 9876 (refer to `karma.config.js` as the configuration file for unit testing)

3.2 Backend tests

To check Plunner’s backend unit tests install PHPUnit 4.* (https://phpunit.de/), refer to `phpunit.xml` as the configuration file for unit testing

4. Already available implementation

An already available implementation of the application exists on the domains admin.plunner.com(for the frontend) and api.plunner.com(for the backend).
To start using the application Sign In or Sign Up from plunner.com.
Tests accounts that shows immediately interesting functionalities of Plunner are available:

- Organization (testInit@test.com, test)
- Member/Planner (testInit, testEmp@test.com, test)

5. Notes

5.1 Backend

- You should insert your name as author in composer file
- To not to perform tests of console tasks, since they can cause problems on Windows systems and they need specific software, set the following env variable DO_CONSOLE_TESTS to false
- exec calls must be enabled in php-cli
- tmp dir permissions needed
- The project is tested only on linux, we don't know the behavior of critical parts (optimisation and CalDAV sync) on other systems