**MikkoTest - Thought-Yards Framework**

**The Design and Approach Document**

Contents

[**1.** **Introduction** 2](#_Toc367098162)

[**2.** **Pre requisite** 2](#_Toc367098163)

[**3.** **Installation** 2](#_Toc367098164)

[**4.** **Components** 4](#_Toc367098165)

[**5.** **Application and Command Usage** 4](#_Toc367098166)

[**6.** **Framework(ThoughtYards)** 9](#_Toc367098167)

[**7.** **Assumption-** 12](#_Toc367098168)

[**8.** **Thanksgiving and References-** 12](#_Toc367098169)

1. **Introduction**

To achieve the task, the design thought was incepted to create the CLI Application which can cater the “Mikko-Test” for salary disbursement through console.

To start with, application has both CLI as well as the Web Version. The app/framework has been conceptualized, designed and implemented to cater the business needs of any organization using core OOPS concepts MVC patterns, Dependency injections using service as a containers, YAML file Parsers for global configurations(supports framework as well as Application), Class Autoloaders using composers, browser based terminal etc. with simplify code yet powerful features.

I preferred to create a new Framework for the “Mikko Test” rather than using readily available in the market. The framework has been named affectionately as “ThoughtYards”. This framework supports “MVC” pattern for CLI and Web. “ThougtYards” would be highly scalable & sustainable over adding new libraries and using them by simple configuration in YAML files and AOP approach.

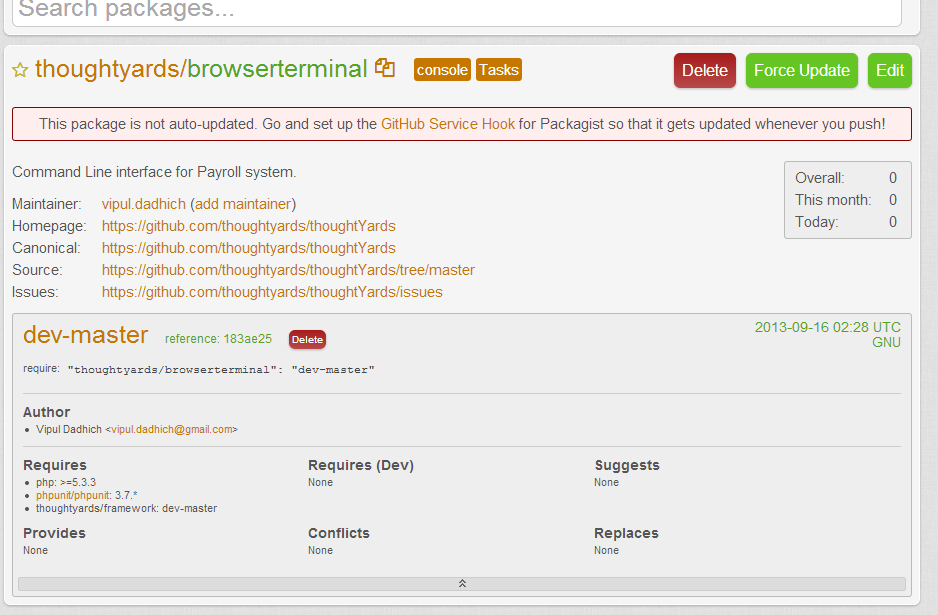
This framework is currently in very nascent stage being it just 7 days old, hence it is not tested on the any aspects and it requires thorough brainstorm to make it production ready. So please do not expects sanity and cleanliness However it works fine for “Mikko Test”.

1. **Pre requisite**
2. Composers – <http://composer.org>
3. GIT Hub – <http://github.com/>
4. PHP 5.3 and greater
5. Access to SSH
6. **Installation**

* Download composer.phar in your root directory
* Copy composer.json attached with email in the folder.
  1. Way 1- Code has been published on gitHub. It can be accessed through

<https://github.com/thoughtyards/thoughtYards>

* 1. Way -2 The code can be auto deployed through <http://packagist.org/> as well



<https://packagist.org/packages/thoughtyards/browserterminal>

$ composer install //Run on root

* It has required dependency on ‘thoughtyards/framework’ which will be downloaded from packagist automatically as it has child dependency.
  1. php 5.3/
  2. "thoughtyards/framework": "dev-master"
* Create the virtual host file and paste the text in the file Vhost file.
* **\*\*You might want to tweak the path for application.**

**Basically you need to load the index.php file from the below path**

<VirtualHost \*:80>

ServerAdmin postmaster@thoughtyards.com

DocumentRoot "public\_html /root/vendor/thoughtyards/browserterminal /"

ServerName thougtyards-mikko. demo

<Directory " public\_html /root/vendor/thoughtyards/browserterminal /"">

Options Indexes FollowSymLinks Includes ExecCGI

AllowOverride All

Order allow,deny

Allow from all

</Directory>

</VirtualHost>

**\*\*\*Note: -Without VHost entry Application will not work properly.**

* Activate the site

/etc/hosts entry

127.0.0.1 thougtyards-mikko.demo

* Make entry in proxy of you browser “Exception to proxy”
* Ready to go……

1. **Components**

* CSV parser
* Browser based terminal aka Shell Commander
* YAML file parser
* Dependency injection using AOP
* Console command kit and WebKit (Integrated with framework)
* Logger

1. **Application and Command Usage**

The application is named as “TerminalKit/ Payroll” and the application source code lies inside “src/Terminalkit/Payroll”.

The application supports browser based terminal which exactly has same behavior of terminal. Since it’s just demonstration purpose and need to go compatibility and test for various scenarios and command. The terminal can be accessed through the server alias:

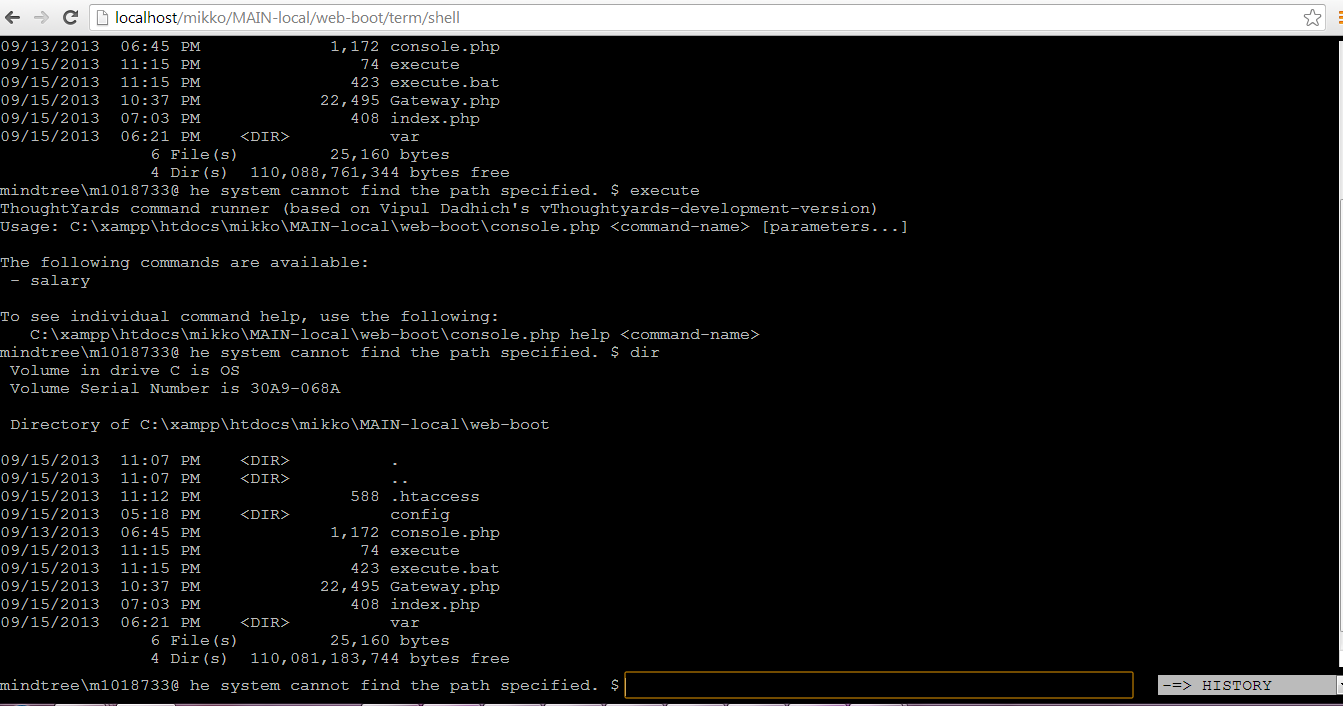
Example:- <http://thoughtyards-mikko.demo/>

This will render the shell in the browser using MVC patterns.

If system prompts you for credentials here are they. But the credentials would only require when you access the command through browser.

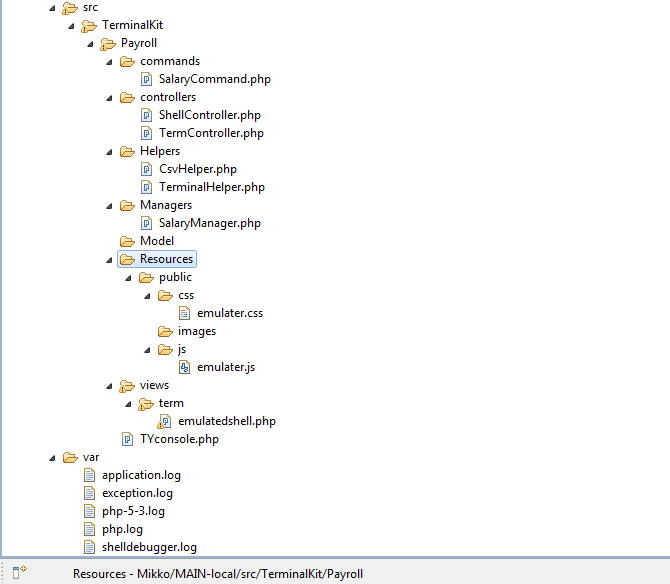
User name: **user**

Password: **pass**

This shell emulator will be used to access all the commands available in terminal kit or running any operating system command. This tool is very beneficial in cases where the user have very limited access to the servers. Through this approach “ThoughtYards” will give full control over the application and shell.

Folder structure of the module “TerminalKit/Payroll”.

**\*\*This does not include the framework structure.**



Explanation of Directory structure:

1. **Controllers**- It is the collection of controllers and actions, rerouted from ThouhtYards kinetic/controller components and other base classes. Behind the scenes framework is taking care of passing headers and converting the request into objects.

The rendering method(s) automatically renders the presentation layer by follow the folder structure rules defined in the class itself and can be modified.

**Note\*\*: Framework will be having XML Parsers which resolve Layout’s files and maintain the dependencies like assets and block related assets followed by theme.**

1. **Commands**- It is the collection of commands and actions, rerouted from ThoughtYards Kinetic/ConsoleKernel components and other base classes. Behind the scenes framework is taking care of passing arguments and converting the request into objects. The generic method(s) automatically renders the output in command line interface and return to shell.
2. **Helper**- Helper consists of helper classes which caters special utilities like CSV parsing and terminal command parsing to generate output in presentation layer.

**Terminal Helper**-This works on Ajax based responses which based on session handlers storage of data. It displays the output of command executed through php in shell commander (Browser based terminal kit).

1. **Managers**- Managers handles the core business logic of the organization. It has been purposely moved from controller to managers to enhance the reusability of the code and make the business logic available across the entire application as and when required through containers as services. This classed could be extended to ‘n’ level and we have have n numbers of command. Managers supported auto loading by simple configuration available in “webboot/config/ AppTerminalWebConfig.php” for WEB and “AppTerminalConsoleConfig.php” for CLI.
2. **Resources** – Resources contain the assets for the web application presentation layer.Assets like CSS,JS and images
3. **Views**-View contain the template file having extension.php.Later when ThoughtYard will move to next level, this presentation may use .twig, .tpl, . phtml .Extensions depend upon which library an organization choose for the presentation pupose.As ThoughtYards fully supports vendor management through pacakegist.
4. **Dependency injection**- These classes can be instantiated using the simple configuration like managers and can be modified to handle event hooks, Inversion of control, error handlers, etc.

**Commands usage for Terminal Kit (Mikko) Application-**

**TerminalLoader**

1. **Controllers**: TerminalController.php
2. **Helpers** - > (used as a dependency Injection) these helpers are loaded as the Services in the Container. That can be used through $this->Gateway()->getComponent and accessd in the program.
3. **Managers**: The Business logic is offloaded to the Managers. Function related to business logic can be accessed from Console as well as CLI.

**SalaryManger.php:**

Reason for offloading Business logic in to managers is for reusability of the Component. Since, we also want the business logic to expose to Web along with Console or anywhere in the program.

The Managers can be available through Containers wherever and whenever required in the application.

**CSVHelpers:**

This is mini utility for CSV parsing, with help of this class we generate the CSV and store it in the directory.

We are sending array of salary dates and it generates the CSV file.

**Exporting the CSV to file.**

The export directory can be changed using booting.yml file. By default I set var/export as the output folder.

**TerminalHelper**

Thanks to Pavel Tzonkov

/\*\* SHELL COMMANDER 1.0

\*

\* **@package** shcmd

\* **@version** 1.0

\* **@author** Pavel Tzonkov [pavelc@users.sourceforge.net](mailto:pavelc@users.sourceforge.net)

\* **@copyright** 2005-2007 Shell Commander Project

\* **@link** <http://sourceforge.net/projects/shcmd>

\* **@license** <http://www.opensource.org/licenses/gpl->

\* license.php

\* GPL

\*/

Code downloaded from this link and modified to fit in the application in object oriented aspects.

This Class keeps no reliability for OS command executions but work good with Mikko test. Like ‘php’ and ‘di’r stuff. This code is developed for demo purpose. This Class is used as the dependency Injection to load the Terminal on runtime in controllers.

I have tried to make the program to fit in to the MVC model.

**Console Application Usage:-**

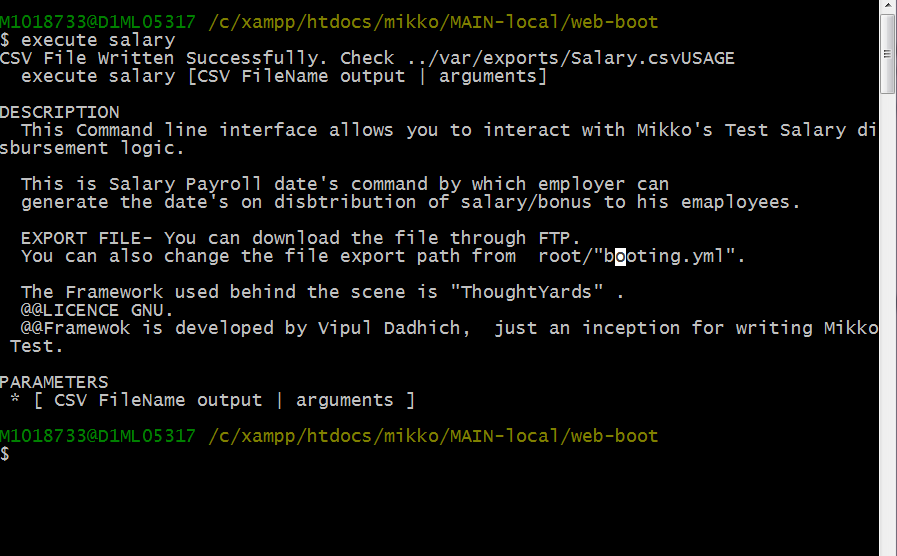
**SalaryCommand.php**

Run($args) method is automatically executed though framework and for business logic we can rely on Mangers.

Ex- getContainer()->get('salary.manager')->getSalary();

Basic command you would like to use on browser Shell.

$ execute salary



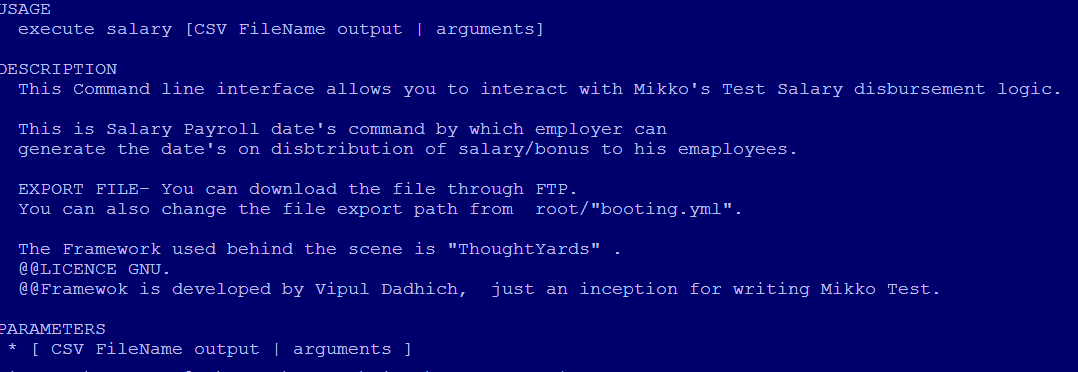
PARAMETERS

[ CSV FileName output | arguments ]

\*\***By default program list down the commands available to get executed. In my case I have only one command name Salary so one command will be listed here. You can add multiple commands by creating new files in commands folder.**

Program also display the basic help and arguments by typing

$ execute help salary



Finally executing the command

$ execute salary Thougtyards.csv

Status message is getting print on shell-

SUCCESS: CSV File Written Successfully. Check ../var/exports/Thoughtyards.csv

To open the file and see what gets print in the file.

$ bin/excel var/exports/ Thoughtyards.csv

\*\*Web based Command to export the csv in var folder.

<http://thoughtyards-mikko.demo/shell/live>

1. **Framework(ThoughtYards)**

Under the hood, behind the scenes “ThoughtYards” working as a backbone to support the structure.

Framework uses light weighted YAML file for the configuration. System configurations can be played around easily by changing values in YAML file inside /web-boot/config/\*. These small changes can bring lot of framework related changes in and this is not common in most of the frameworks.

Example: Booting.Yml can control the demonstrated controlling Logging of the application.

booting:

logging: true

developer\_mode: true

log\_warning\_level: 2

log\_dir: var\

This is really helpful on production level application.

1. **YAML** - > Application supports these configurations for now.

const APP\_CONFIG='config\_%s';

const APP\_ROUTERS='router\_%s';

const APP\_VALIDATIONS='validation\_%s';

const APP\_BOOT='boot\_%s';

Usage: getConfig($type=false, $fileName=false)

**NOTE:- Do not use any “,” to separate the configurations sign in the yaml file.**

1. **Class TYCofig.php**

Booting.yml holds the configuration for the entire application:

app:

config\_dir: /web-boot/config (is pointing to the configuration directory)

Configurations of the application can be accesses using this method

Gateway::getAppConfig();

**Usage**:- TYConfig::tobject is use to get the Configurations in the object form.

1. **Global Registries**

Provision for storing the Variable in the registry in Gateway class.

1. **Application Components**
   1. Application Logs:-

Logs directory can be specified in the booting.yml file.

Note: This is the Framework booting configurations. Defined configurations will be available across the application by calling ThoughtyardsConfiguration Class.

*log*($msg,$level,$category='application');

generates var/ directory having file “application.log” by default.

You can also change the level of the warning logs and file for generating the error trace.

Usage:- **Gateway**::*trace*('Custom message get attached to trace',2, ‘filename’);

Sample Log:-

Loading "request" application component

Backtrace

in C:\xampp\htdocs\mikko\MAIN-local\web-boot\Gateway.php (351)in C:\xampp\htdocs\mikko\MAIN-local\vendor\thoughtyards\framework\src\Kinetics\Component\TYModule.php (391)in C:\xampp\htdocs\mikko\MAIN-local\vendor\thoughtyards\framework\src\Kinetics\Component\TYApplication.php (526)in C:\xampp\htdocs\mikko\MAIN-local\vendor\thoughtyards\framework\src\Kinetics\HttpKernel\WebKit.php (552)in C:\xampp\htdocs\mikko\MAIN-local\vendor\thoughtyards\framework\src\Kinetics\Component\CApplication.php (179)in C:\xampp\htdocs\mikko\MAIN-local\web-boot\Gateway.php (239)in C:\xampp\htdocs\mikko\MAIN-local\web-boot\index.php (17)

Default logging file:-

application.log prints the Backtraces.

exception.log all ThoughtYards Exception get logged in the exception.

**Dependency Injection**

ThoughtYards\Fixtures\DependencyInjection\IOC;

Usage:-

Gateway::*app*()->getContainer()->get('csv');

**RequestHandler**

Handle the Request coming to controllers of the Application.

**ResponseHandler**

Handle the Response going to presentation of the Application.

**GateWay**

$object= $this->getGateWayApp()->getContainer()->get('csv\_helper');

**Containers**

You need to define the dependency in the AppterminalConfig.php

Like

'container' => **array**(

'class' => 'TYContainer',

'services' => **array**(

'csv\_helper'=>'\TerminalKit\Payroll\Helpers\CsvHelper'

)

1. **Assumption-**

* User uses Windows and Linux operating systems
* This application is basically runs on the PHP CLI,
* This application runs on the framework designed and architected by Vipul i.e <ThoughtYards>
* The Docs for the Framework will be provided

1. **Thanksgiving and References-**

* <http://sourceforge.net/projects/shcmd/?source=dlp>
* <http://www.yiiframework.com/>
* <http://stackoverflow.com/>
* <http://php.net/>
* <http://www.thoughtyards.info/>
* <http://www.thoughtyards.com/>