

# **TECHNICAL DOCUMENTATION FOR JEJ\_MIS 0.5.XX**

June 24, 2015

Author: Corsanes, John Cyril C.

[johncyrrilcorsanes@gmail.com](mailto:johncyrrilcorsanes@gmail.com)

Software Engineer from THE BIG FIVE

## TABLE OF CONTENTS

1. Abstract .....	3
2. Introduction .....	3
3. Objectives .....	3
4. Features .....	3
5. System Requirements .....	4
6. Getting Started .....	5
7. Conclusion .....	10
8. Changelog .....	10
9. Appendix .....	11

# JEJ\_MIS TECHNICAL INFORMATION FOR 'THE BIG FIVE' AND FUTURE DEVELOPERS

## Abstract

The JEJ\_MIS is the Management Information System for Company we've been proposing (Confidentiality). This system is a Web Science study using the latest technologies such as PHP, HTML and MySQL. The study is based on my Web Science Research with Objectives, Scopes and Limitations by the past few chapters of our (THE BIG FIVE) study.

The JEJ\_MIS was made with [MINI](#) and [PHP-LOGIN 2.1.1](#) by [panique](#). This was inspired from MVC (Model-View-Controller) Concept and Microsoft's MVC Web Framework. We've combined the two frameworks into one and the results are good as usual.

## Introduction

Many technologies rapidly evolved in every second of our lives. Web Science is already involved all over the world such as using Social Media (like Facebook, Twitter, Tumblr), Streaming Multimedia Content and even creating projects without paying for proprietary programs and applications.

Also the industry rapidly evolving every year, even every quarter of the year and it occurs constraints and problems for those organizations which system should they use in the industry. Many Organizations aware and finding the suitable system that is reliable, understandable, feasible, accessible and user-friendly.

Major IT Companies like IBM, MICROSOFT, ORACLE, and CISCO leads in the IT industry today. Some IT people does not need to learn anything from complex standards if there is more secured and easy to use for developers, software engineers, and system analysts. Open-source Technologies is one of the leading technologies today because it's free to install and use, and we've decided to develop a Management Information System for Company we've proposed using the Open-Source technologies such as PHP and MySQL.

## Technical Objectives

The technical objectives for this system is to ensure the functionality, stability, compatibility, security and user-friendliness.

## Technical Features

Here some proposed and to be features to propose.

- Using MVC (Model-View-Controller) skeleton
- Inspired from Microsoft ASP.NET Web framework
- No extra syntax required to learn (Unlike CakePHP and CodeIgniter)
- Multi-Layered Security features (especially login and CRUD actions)
- Mobile Friendly UI
- Backward Compatibility with older PHP versions

## System Requirements

Here's the Recommended Technical Requirements for this system.

### End-User Requirements

- **CPU/Processor:** Any Quad Core Intel or AMD 64-bit Processor
- **RAM:** 4-8GB DDR3
- **Hard Drive:** At least 300GB of HDD/SSD (SATA or IDE)
- **Network:** Any internet connection at least 3mbps
- **Operating System:** Any OS such as Windows/Linux/Mac OS X, etc.
- **Others:**
  - XAMPP/WAMPP is required with all installed components
  - NetBeans 8.0.2 and up
  - Composer
  - Git Module (for updates/maintenance)

### End-User Requirements for Mobile

#### For iOS Devices

- **Devices:** At least iOS 7 phone/tablet or higher

#### For Other Devices

- **Operating System:** At least Android 4.1 Jelly Bean or higher

### Server Requirements

- Web Hosting for CRM
- Same as End-User except the Operating System
- Linux – any LTS from 2014



## Getting Started (For Administrators)

1. Make sure that you are running XAMPP with Apache and MySQL Servers running.
2. Download the ZIP file or pull from master on [jejMIS repository](#).
3. Make sure the repository is on XAMPP's htdocs directory with the new folder (name is optional)
4. Go to MySQL Admin button in XAMPP Control Panel.
5. Make sure that your database is filled with users. (SQL file coming soon)
6. Go to your application and modify settings in  
\*\*/htdocs/\*your new app\*/cpanel/application/config/config.php
7. Now run the application (e.g. http://localhost/\*your folder name in htdocs\*/)

How the MVC Framework skeleton works?

1.) This .htaccess file will load the redirect into public using RewriteEngine (the root)

Name	Date modified	Type	Size
_install	3/29/2015 5:38 PM	File folder	
application	3/25/2015 4:39 PM	File folder	
public	4/16/2015 2:04 PM	File folder	
vendor	3/25/2015 4:39 PM	File folder	
.htaccess	3/24/2015 7:28 AM	HTACCESS File	1 KB
composer.json	3/24/2015 7:28 AM	JSON File	1 KB
composer.lock	3/24/2015 7:28 AM	LOCK File	8 KB

2.) This /public directory will start the application using the index.php inside

*This screenshot is based in CPANEL Edition / beta version and it may vary to the Version 0.5.XX*

### Directory info:

- /\_install – DB Installation
- /application – The main application
- /vendor – 3<sup>rd</sup> Party packages from composer

## Accessing the Main Application

As we can see this code below in /public/index.php. This is the starting point of the application.

```
<?php
2
3 /**
4  * jejMIS - Management Information System
5  * Modified from PHP-LOGIN 2.1.1 and MINI by panique
6  *
7  * @package jejMIS
8  * @author jccultima
9  * @link https://github.com/jccultima123/jej\_mis/
10 *
11 */
12
13 /**
14  * THIS IS THE STARTING POINT OR ROOT OF THE MVC FRAMEWORK
15  * This will load all libraries and configurations required
16 */
17
18 // TODO get rid of this and work with namespaces + composer's autoloader
19
20 // set a constant that holds the project's folder path, like "/var/www/".
21 // DIRECTORY_SEPARATOR adds a slash to the end of the path
22 define('ROOT', dirname(__DIR__) . DIRECTORY_SEPARATOR);
23 // set a constant that holds the project's "application" folder, like "/var/www/application".
24 define('APP', ROOT . 'application' . DIRECTORY_SEPARATOR);
25
26 // This is the (totally optional) auto-loader for Composer-dependencies (to load tools into your project).
27 // If you have no idea what this means: Don't worry,
28 if (file_exists(ROOT . 'vendor/autoload.php')) {
29     require ROOT . 'vendor/autoload.php';
30 }
31
32 // load application config (error reporting etc.)
33 require APP . '/config/config.php';
34 // other configs pulled from PHP-LOGIN
35 require APP . '/config/autoload.php';
36
37 // FOR DEVELOPMENT: this loads PDO-debug, a simple function that shows the SQL query (when using PDO).
38 // If you want to load pdoDebug via Composer, then have a look here: https://github.com/panique/pdo-debug
39 require APP . '/libs/helper.php';
40
41 // other libs pulled from PHP-LOGIN
42 require APP . '/libs/Auth.php';
43 require APP . '/libs/Session.php';
44 //DISABLED FOR NOW FOR SPEED
45 //require APP . '/libs/password compatibility library';
46
47 // load application class
48 require APP . '/core/application.php';
49 require APP . '/core/controller.php';
50
51 /**
52  * This will start the application
53  */
54 $app = new Application();
55
```

This line will initialize the application directory instead of the index.php will initialize

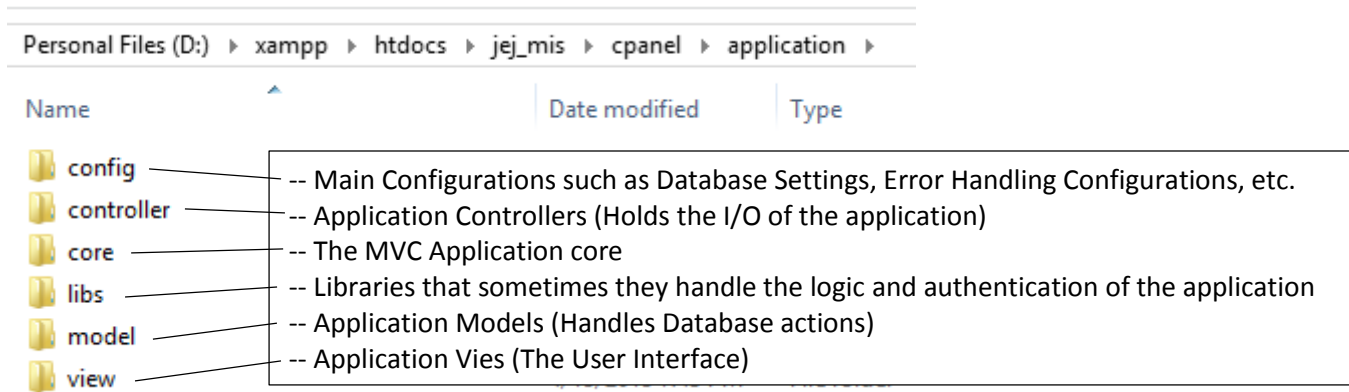
This 'APP' variable can change at any time but this will be the main role of the application

The 'require' function will load the MVC skeleton including public and private functions (please review the codes inside this file)

This line will start the MVC Framework and the application core

*This screenshot is based in 0.2 and it may vary to the Version 0.5.XX*

## Accessing the Main Application



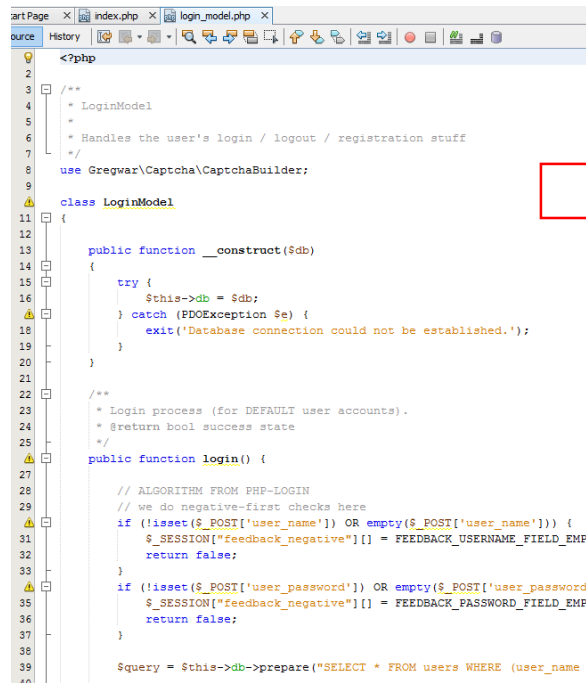
This is the MVC Structure of the Main Application

## Creating Application Features/Page

By creating a page or by adding a feature, the developer must presence the three needs for this application, The Model – View – Controller concept. If one of them misses, the application might occur unknown errors.

### The Model

The Model will handle the query actions to the database. The model can do everything to the database using the PDO Controller. PDO is widely used in large-scale systems and it's suitable in this application. PDO can do the CRUD (Create, Rename, Update, and Delete) actions. If you are creating a page without any feature (i.e. about page), the model might not be needed. But if you are creating a feature outside the home page, you must call a model or instantiate it in /core/controller.php.

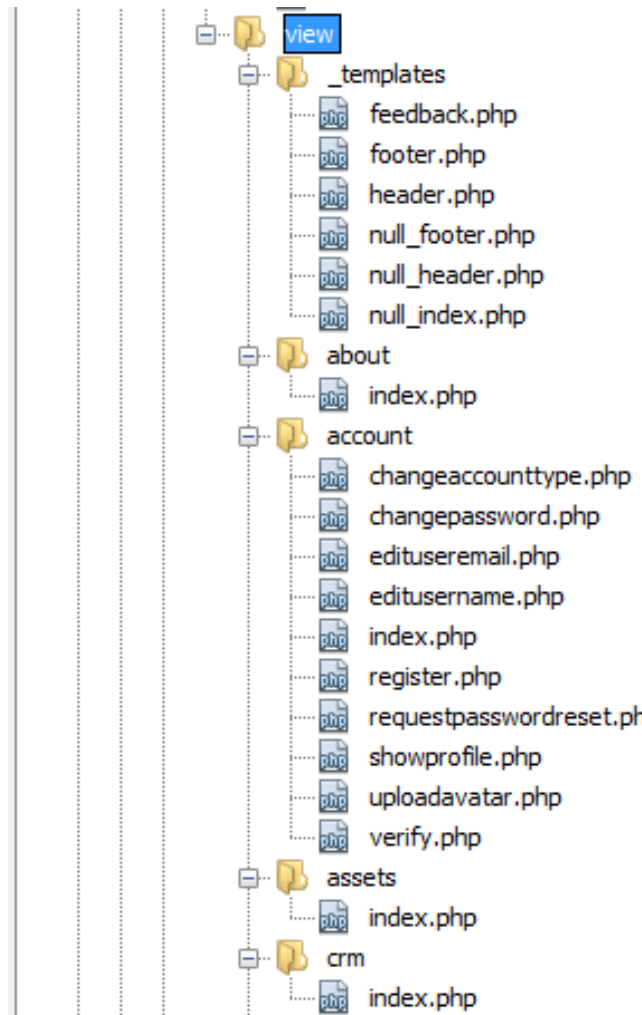


The login\_model.php file in Model directory



## The View

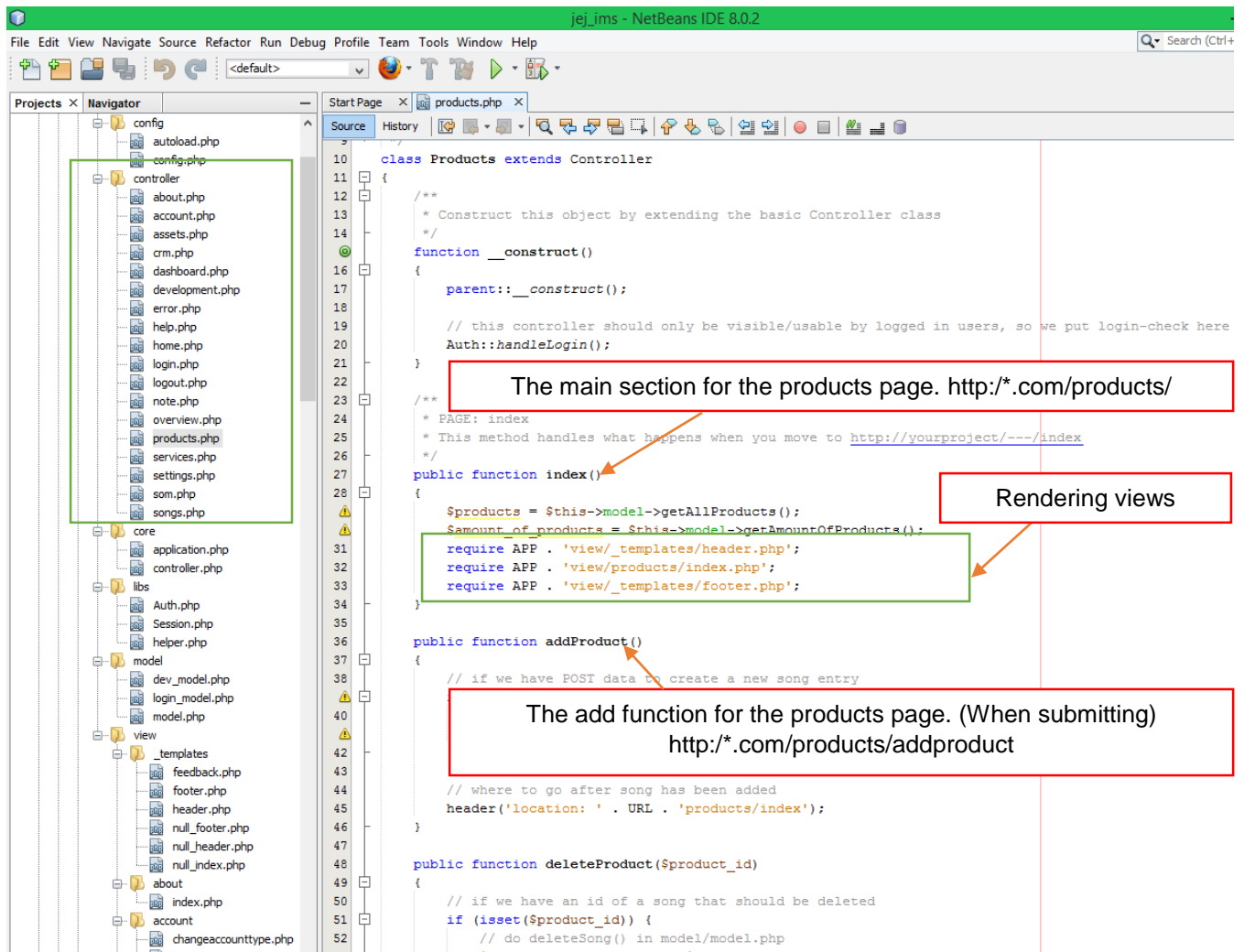
When creating a view, you must finish at least one page with 3 files, the header, content, and the footer. To make it cleaner, the author of this application prefer that make the default header and footer in `_templates` in Views directory. Here's the sample.



*This screenshot is based in 0.2 and it may vary to the Version 0.5.XX*

## The Controller

The controller handles the logic (Input/output of the user such as submitting forms) and Authentications of the Application. (NOTE: The model does not connected to the Controller and View based on the Microsoft's MVC Concept.). In this application, the developer must encode the “public function index()” which is it's an default index of the page. But if you want to create a sub-page, you can create a public function in current controller you've been editing, but this time, the name of the function must be also in the sub-page. Here's the example.



*This screenshot is based in 0.2 and it may vary to the Version 0.5.XX. PRODUCTS may be not included in the final product.*

## Conclusion

Open-Source projects might not be good but there's no big deal about it. Still, the technology we have today especially on the web/internet would be the way to improve of our lives and also to the industry. We are hoping that this might be using in the future for the next generations.

## Changelog

### VERSION 0.5.11 – June 24, 2015

- Added Bootstrap UI Framework
- Revamped UI by separate .css
- Various MVC patches
- Added form validation via nod.js
- Fixed Captcha
- Updated JQuery
- User Authentications with Administrator Mode
- Added Verification Mail including for Password Reset
- Mobile UI patches
- Fixed git ignore lists
- CRUD Patches
- Added Environment Detection
- Added Debug Info when it's under Development
- Added Browser Detection if it's compatible or not
- Separated more than one Base Controllers for User Level Authentication
- Combined Login Function (Since it can detect if it's an admin or not)
- Added Caching Objects using from Web Server (may not compatible with IIS)
- Added DB Relationships
- And much more...

### VERSION 0.2 – April 27, 2015

- Added Other Models
- Improved UI
- Changed Title
- Cleaned Redundant Codes
- Fixed CRUD Database Actions
- Fixed MVC Core
- Updated Project Information

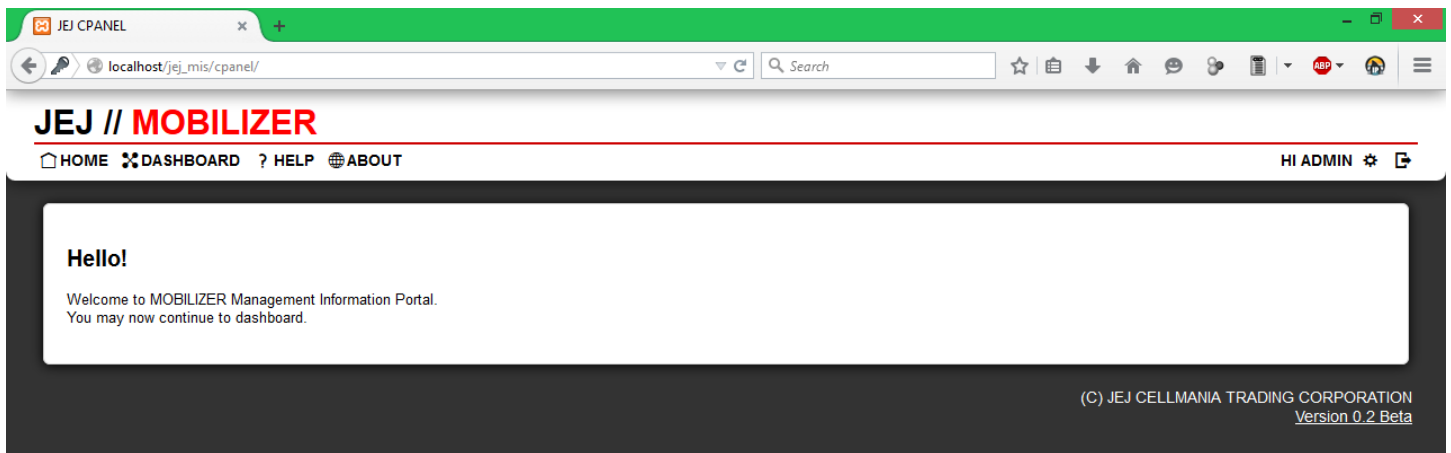
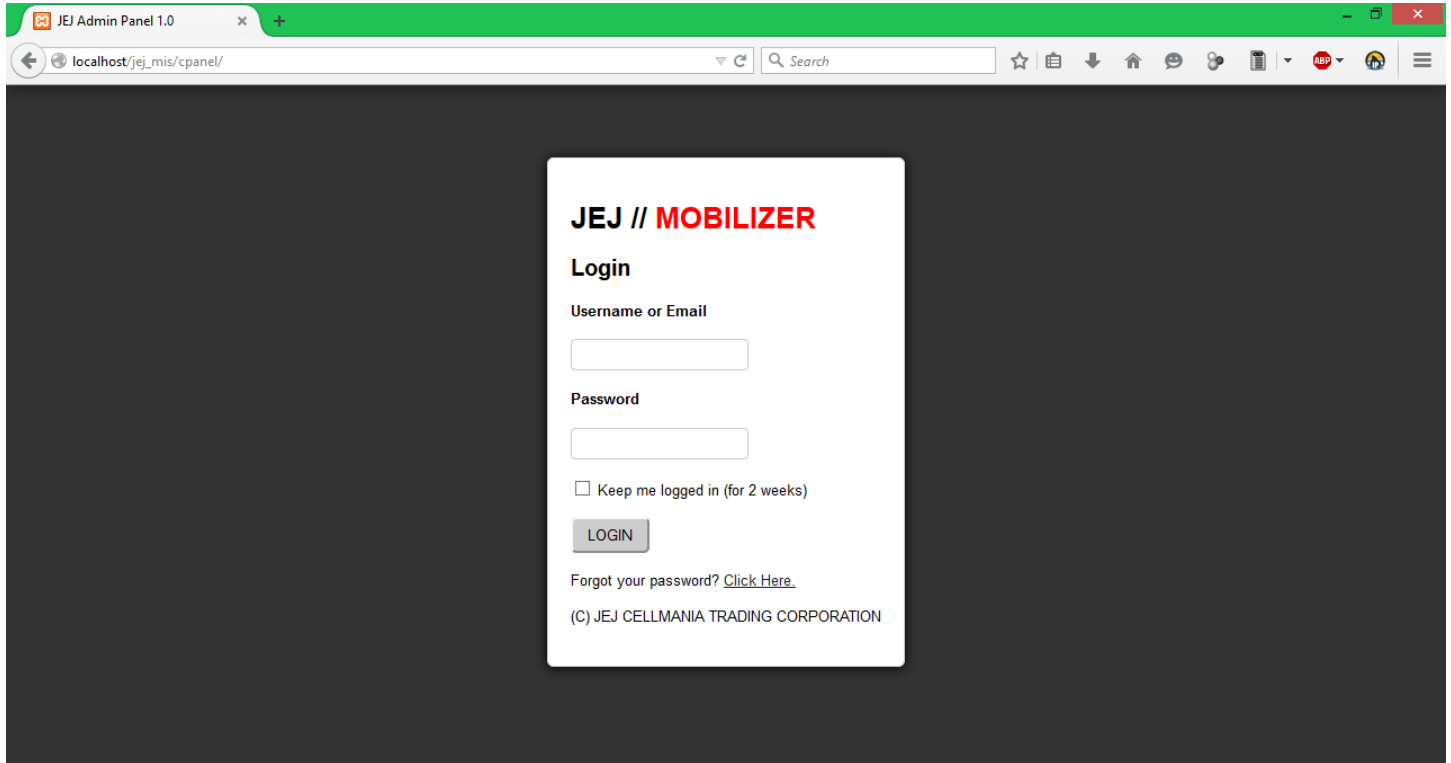
## VERSION 0.1.9 – April 15, 2015

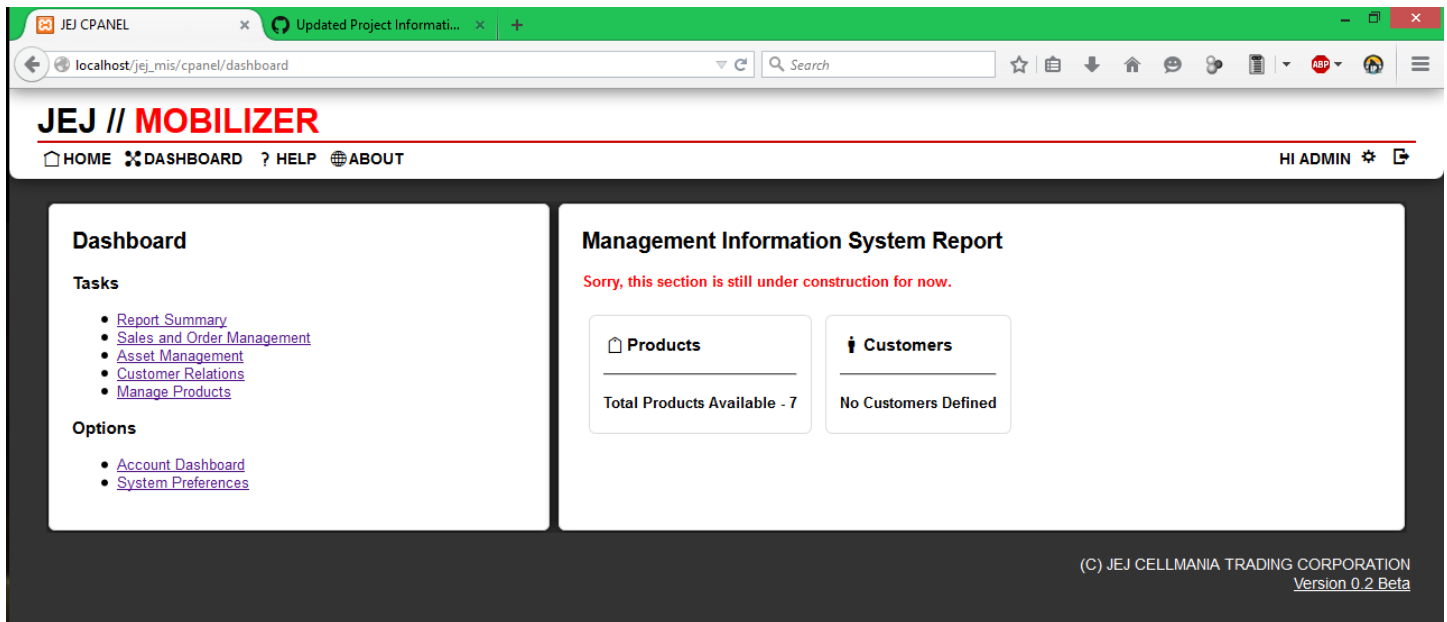
- Added Miscellaneous Libraries
- Added JQuery
- Dumped Application for Public Site
- Revamped UI for Speed and Accessibility
- Optimized Models
- More Secured UI, Inspired by JRU AIMS Student Management
- Added CSS Bootstrap
- Arranged files for /public folder
- Some Changes in Authentications
- Improved Error Detection
- Performance Improvements

Other Changes already in [https://github.com/jccultima123/jej\\_mis/commits](https://github.com/jccultima123/jej_mis/commits)

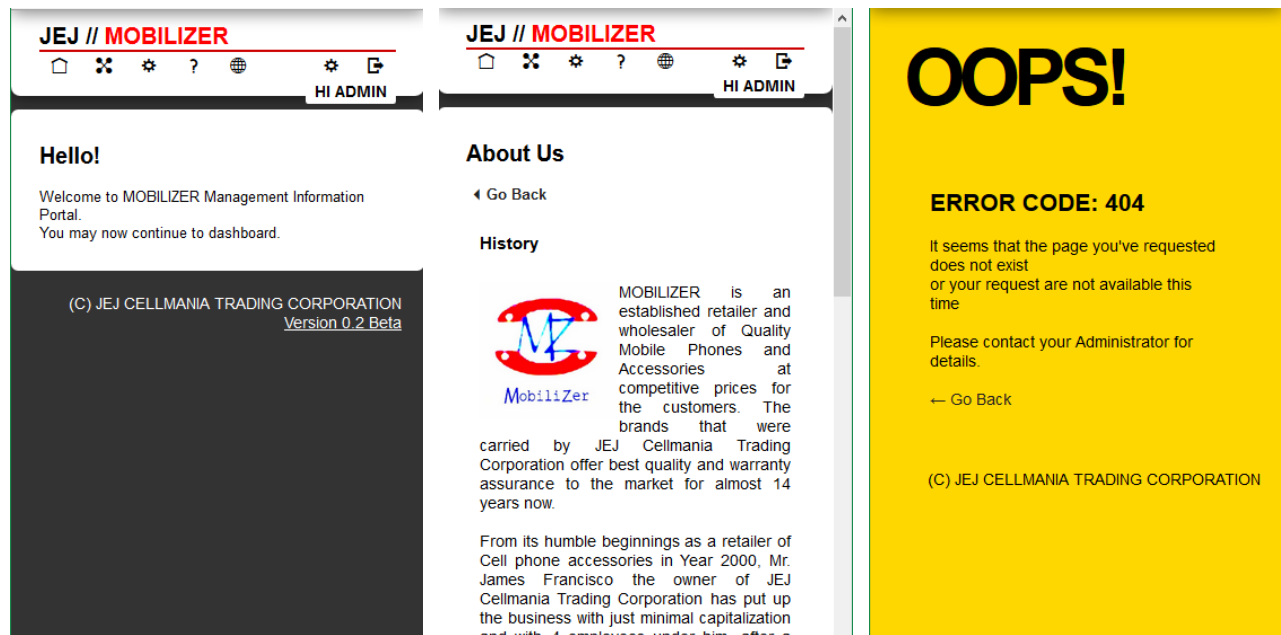
## Appendix

### Running in Windows 8.1 64-bit





### Desktop Version



### Mobile Version

More Information:

jejMIS Repository Site: [https://github.com/jccultima123/jej\\_mis](https://github.com/jccultima123/jej_mis)