# Coding Standards Document

## Laravel 5.2

### PHP, JavaScript

This document contains the coding standards, naming conversions, comments and coding styles regarding the application, University Resource Scheduler.

This application use PSR-1 and PSR-2 Standards developed for PHP. Laravel Autoloader use PSR-4

## Naming Conversions.

## PHP

### Classes

Class names must always start with capital letters and each new word should start with capitol letter in the class name. All class names must follow Studly Caps.

**class** AdministratorOptionsController **extends** Controller

### Methods

Method names must start with a lowercase letter and each new word proceeding must begin with a capital letter with no spaces. Method should contain a clear idea about the functions’ purpose.

**public function** masterReset()  
{

//Your Code

}

### Variables

Variable names should follow camel case format defined by PSR1. The first letter should be simple while every other new word preceding after that must be a capitol letter without spaces.

$variableName

### Controllers & Models

Controllers must always start with a capitol letter and each new word proceeding must start with a capitol letter.

Models must always start with a capitol letter. And each new word preceding should start with a capitol letter.

## JavaScript

### Methods

Methods should always start with a capitol letter and each new word in the method should start with a capitol letter.

**function** MasterReset()  
{

//Your Code

}

### Variables

Variable names should start with a lower case letter while each new word preceding must start with a capitol letter. Variable names should correspond to the value it must hold.

timeSlotFromDatabase

Object variable (prototypes) names should start with Capitol letter and each new word proceeding must start with a capitol letter.

***LecturerTimeData***

## Comments.

Both PHP and JavaScript methods should contain block comments. Comments inside methods should be single line comments. Block level comments should be generated using PHP unit. With a description and the input and output parameters.

*/\*\*  
 \* Deletes Deadlines  
 \*   
 \** ***@param*** *Deadline $id  
 \** ***@return*** *\Illuminate\Http\RedirectResponse  
 \** ***@throws*** *\Exception  
 \*/***public function** DeadlineDelete(Deadline $id)  
{  
 //Your code  
}

JavaScript comments should also be block comments with an explanation on what the method should do.

*/\*\*  
 \* Create XML header for Excel File  
 \* Exports the timetable in excel format  
 \*/***function** *ExportXLS*()

{

//Your Code

}

Single line comments should be used when or if a part of the code need further explaining. Should use the ‘//’ for both JavaScript and PHP

//Your Code

## Indentation and Layout

Indentation must be performed using a TAB character. Spaces are not allowed. These standards apply for both PHP and JavaScript.

### Conditions and loops.

If conditions and loops must use the [Allman](https://en.wikipedia.org/wiki/Indent_style#Allman_style) layout style.

**if**($condition)  
{  
 //Your Code  
}  
**else**{  
 *// Your Code*  
}

**for**()  
{  
 //Your Code  
}

### Assignment

When assigning values, the following assigning style should be used to make it easier to identify and easily read.

$object = **new** Object();  
$object->**value1** = $request[**'value1'**];  
$deadline->**value2** = $request[**'value2'**];  
$deadline->**value3** = $request[**'value3'**];

### Dependency Injection and Variable passing in methods.

### 

Values passed into methods must not contain a space between variables. Each variable should be separated with a comma.

**public function** editUserUpdate(Request $request,User $staff\_id)  
{

//Your Code

}

# References

http://www.php-fig.org/, n.d. *http://www.php-fig.org/.* [Online]   
Available at: http://www.php-fig.org/psr/psr-1/

http://www.php-fig.org, n.d. *http://www.php-fig.org.* [Online]   
Available at: http://www.php-fig.org/psr/psr-2/