HACETTEPE UNIVERSITY DEPARTMENT OF COMPUTER ENGINEERING
BBM487: SOFTWARE ENGINEERING LABORATORY

CODING STANDARD

GROUP – 6

CEMAL ÜNAL	21328538
İREM KOCABAŞ	21328188
METİN ARSLANTÜRK	21426625

CODING STANDARDS

1. PHP Coding Standards

We have used PHP for our implementation of the Library Book Loan System so we have considered PHP coding standards during this process. For that reason we had searched PHP coding standards on the internet and the most detailed and specific one is this one which can be found through link: https://framework.zend.com/manual/1.12/en/coding-standard.html

2. PHP File Formatting

2.1. Indentation

Coding standard indicates: Indentation should consist of 4 spaces. Tabs are not allowed.
We used Sublime Text for code writing platform and it adjusts by itself.

2.2. Maximum Line Length

Coding standard indicates that maximum line length is 80 character but in some special cases 120. We have written considering 80 character limit.

2.3. Line Termination

Coding standard indicates: Line termination follows the Unix text file convention. Lines must end with a single linefeed (LF) character. Linefeed characters are represented as ordinal 10, or hexadecimal 0x0A.
Unix linefeed is "\n" so we used this linefeed on line terminations as well as html linefeed which is "<br/>" tag.

3. Naming Convention

3.1. Classes

Coding standard indicated: Class names may only contain alphanumeric characters. Numbers are permitted in class names but are discouraged in most cases. Underscores are only permitted in place of the path separator.

But we haven't used any classes in our implementation so we didn't need that standard.

3.2. Abstract Classes

Coding standard indicates: In general, abstract classes follow the same conventions as classes, with one additional rule: abstract class names must end in the term, "Abstract", and that term must not be preceded by an underscore.

We haven't used abstract classes also. So as in classes it is not a considerable standard for us.

3.3. Interfaces

Coding standard indicates: In general, interfaces follow the same conventions as classes, with one additional rule: interface names may optionally end in the term, "Interface", but that term must not be preceded by an underscore.

We didn’t need interfaces too. So we didn’t use this standard.

3.4. Filenames

Coding standard indicates: For all other files, only alphanumeric characters, underscores, and the dash character ("-") are permitted. Spaces are strictly prohibited.

In our implementation we only used alphanumeric characters for filenames.

3.5. Functions and Methods

Coding standard indicates: Function names may only contain alphanumeric characters. Underscores are not permitted. Numbers are permitted in function names but are discouraged in most cases. Function names must always start with a lowercase letter. When a function name consists of more than one word, the first letter of each new word must be capitalized.

Since we have written our codes thinking different files for every page and MVC structure we didn’t need to use neither functions nor methods.

3.6. Variables

Coding standard indicates: Variable names may only contain alphanumeric characters. Underscores are not permitted. Numbers are permitted in variable names but are discouraged in most cases. Variable names must always start with a lowercase letter and follow the capitalization convention.

We have used alphanumeric characters in our implementation only. We also considered the capitalization convention in most of our codes.

3.7. Constants

Coding standard indicates: Constants may contain both alphanumeric characters and underscores. Numbers are permitted in constant names.

In our implementation we haven’t used any constant.

4. Coding Style

4.1. PHP Code Demarcation

Coding standard indicates: HP code must always be delimited by the full-form, standard PHP tags: <?php …?>.

All the PHP codes in our implementation has placed between these tags.

4.2. Strings

4.2.1 String Literals

Coding standard indicates: When a string is literal (contains no variable substitutions), the apostrophe or "single quote" should always be used to demarcate the string.

We have used apostrophe to demarcate the string literal in all of our files.

4.2.2 String Literals Containing Apostrophes

Coding standard indicates: When a literal string itself contains apostrophes, it is permitted to demarcate the string with quotation marks or "double quotes". This is especially useful for SQL statements.

Since all the date we have used are kept in MySQL database we had to consider this standard in every PHP file in the Controller.

4.2.3. Variable Substitution

Coding standard indicates: Variable substitution is permitted using either of these forms:
$greeting = “Hello $name, welcome back”;
$greeting = “Hello {$name}, welcome back”;

We have used the first form in every file.

4.2.4. String Concatenation

Coding standard indicates: Strings must be concatenated using the "." operator. A space must always be added before and after the "." operator to improve readability.

String concatenation have used in a lot of our files so we have used this method.

4.3. Arrays

4.3.1. Numerically Indexed Arrays

Coding standard indicates: An indexed array may start with any non-negative number, however all base indices besides 0 are discouraged.

We haven’t used numerically indexed arrays in our implementation.

4.3.2. Associative Arrays

Coding standard indicates: When declaring associative arrays with the Array construct, breaking the statement into multiple lines is encouraged.

We haven’t used associative arrays in our implementation.

4.4. Classes

4.4.1. Class Declaration

Coding standard indicates: Only one class is permitted in each PHP file. Placing additional code in class files is permitted but discouraged. In such files, two blank lines must separate the class from any additional PHP code in the class file.

We haven’t used any classes in our implementation.

4.4.2. Class Member Variables

Coding standard indicates: Any variables declared in a class must be listed at the top of the class, above the declaration of any methods.

We haven’t used any classes in our implementation.

4.5. Functions and Methods

Coding standard indicates: Methods inside classes must always declare their visibility by using one of the private, protected, or public modifiers. As with classes, the brace should always be written on the line underneath the function name. Space between the function name and the opening parenthesis for the arguments is not permitted.

Since we have written our codes thinking different files for every page and MVC structure we didn’t need to use neither functions nor methods.

4.6. Control Statements

4.6.1. If/Else/Elseif

Coding standard indicates: Control statements based on the if and elseif constructs must have a single space before the opening parenthesis of the conditional and a single space after the closing parenthesis. Within the conditional statements between the parentheses, operators must be separated by spaces for readability. Inner parentheses are encouraged to improve logical grouping for larger conditional expressions. The opening brace is written on the same line as the conditional statement. The closing brace is always written on its own line. Any content within the braces must be indented using four spaces.

We have used if/else/elseif statements quite often. Because of that we have written the statements in our codes considering these standards about parenthesis placements.

4.6.2. Switch

Coding standard indicates: Control statements written with the "switch" statement must have a single space before the opening parenthesis of the conditional statement and after the closing parenthesis.

We haven’t used switch statement in our code.