Mawlana Bhashani Science and Technology University

Department of Information and Communication Technology

Course Code : ***ICT-3210***

Course Name : ***Software Engineering Lab***

Name of the Lab : ***Model, View and Controller***

Lab No. : ***05***

Submitted to:
Mr. Tanvir Rahman
Lecturer
Dept. of ICT, MBSTU
Santosh, Tangail-1902

Submitted by:
Md. Nazmul Hasan
ID: IT-17005
Session: 2016-17
3rd Year 2nd Semester

Date of Submission: 31-October-2020

Model, View and Controller of "Laravel"

Model: It belongs to all the data related logic with which the user works. It can be used to represent:

- Data that is being transferred between the View and Controller components
- Business logic related data

For example, a Customer object grab the customer information from the database then manipulate it and update it data back to the database.
To get started, let's create an Eloquent model. Models typically live in the app\Models directory, but you are free to place them anywhere that can be auto-loaded according to your composer.json file. All Eloquent models extend Illuminate\Database\Eloquent\Model class.

The easiest way to create a model instance is using the make:model Artisan command:

```
php artisan make:model Student
```

If you would like to generate a database migration when you generate the model, you may use the --migration or -m option:

```
php artisan make:model Student --migration

php artisan make:model Student -m
```

View: The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.

Views contain the HTML served by your application and separate your controller/ application logic from your presentation logic. Views are stored in the resources/views directory. A simple view might look something like this:

```
<!-- View stored in resources/views/welcome.blade.php -->

<html>
    <body>
        <h1>Hello, {{ $name }}</h1>
    </body>
</html
```

Since this view is stored at resources/views/welcome.blade.php, we may return it using the global view helper like so:

```
Route::get('/', function () {
    return view('welcome', ['name' => 'Ruhan']);
});
```

Controller: Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.

Below is an example of a basic controller class. Note that the controller extends the base controller class included with Laravel. The base class provides a few convenience methods such as the middleware method, which may be used to attach middleware to controller actions:

```
<?php

namespace App\Http\Controllers;

use App\Http\Controllers\Controller;
use App\Models\User;

class UserController extends Controller
{
    /**
     * Show the profile for the given user.
     *
     * @param  int  $id
     * @return \Illuminate\View\View
     */
    public function show($id)
    {
        return view('user.profile', ['user' => User::findOrFail($id)]);
    }
}
```

You can define a route to this controller action like so:

```
use App\Http\Controllers\UserController;
Route::get('user/{id}', [UserController::class, 'show']);
```

Conclusion: From this Lab, I've come to learn the Model, View and Controller of Laravel framework. All the three components are very important for any application as it handle all the specific development aspects of that application. MVC is amongst the most used industry-standard web development framework for the creation of scalable and extensible projects.