Abstract

Abstract With rapid advancement in the field of computer science, the ways we use and interact with web applications have changed immensely. Developers must create web applications for browsers, cell phones, and search engines that are accessible and easy to use in various devices. Therefore, the efficiency of software development is critical. Software Design Patterns are an essential part of software development which is intended to solve real-world problems by creating templates of best practices.

Design patterns bring clarity, cost-effectiveness, and better communication in the software development cycle. They also improve the development speed, support features, and usage, and they reduce expenses. Documentation and maintenance of established web applications frameworks are major advantages of software design patterns. Which is Model-View-Controller (MVC) software design patterns. It analyzes Laravel PHP web application development frameworks. MVC facilitates reuse of code and separation of application layers. It explains the development experience of Plan Management Web application on Laravel.Web applications use compatible programming languages such as HTML, JavaScript, and CSS. Comparative analysis has been done based on the developer's experience and performance monitoring tools.

The Documentation concludes that the Starategic Plan Management System (SPMS) web application built using Laravel on Windows is perform A lot Of Activities and Evaluate users based on their given activity.The developer's conclusion is based on the use of the MVC design pattern, learning curve, framework features, documentation, and application performance.

Acknowledgement

We would like to sincerely thank my advisor MrDaniel . For providing support and feedback throughout of this project. We would also like to thank our friends Amen and Aman for them support and advise,for providing motivation to complete this project.

A big thank you to our family and friends.

Chapter I: Introduction

Software design patterns are reusable solutions to software design problems that recur in application development (Bisson, 2017). They are templates of formalized best practices in designing real-world applications and system.

Developing software applications is complex and costly. Developers need to focus on the behavior and environment of each platform, technology, and architecture. Software Design Patterns help identify potential problems before implementation and reduce code complexity and code clutter.

There are various web application frameworks following many software design patterns. Among them, the developer studies the Model-View-Controller (MVC) pattern. The developer presents the analysis, design, and implementation of software applications based on MVC. MVC offers the possibility of code reuse and provides strict separation of the application layers. MVC gives more control to developers and helps to build lightweight web applications.

Important Definitions

Following are relevant terms defined within the scope of this Report:

AdminLTE : Open-source dashboard & control panel theme that can be used as a template for the front end of the web application.

Apache HTTP server: A free and open-source cross-platform web server.

Blade: A template which is used to apply standard settings to views for the development of the web pages in Laravel framework (Bautista, 2012).

Bootstrap: An open-source and an efficient front-end framework used for web application development. It includes CSS, HTML, and JavaScript in its framework.

Cascading Style Sheets (CSS): CSS is the standard markup language used for presenting the styles of the content in the web pages.

Client-side programming: Also known as front-end programming, includes everything that a user sees and interacts with on a web browser.

Composer: A package dependency resolver for the Laravel web application framework. It is used to manage dependency in PHP.

Cross-Site Reference Forgery (CSRF): CSRF is an attack forcing unwanted actions from the users without their consent.

Database: It is a set of data which is stored in a structured way. The data set can be read/updated/deleted.

Development environment:

The developer develops software in the development environment.

Eloquent Framework: Laravel 5 uses Eloquent as an Object Relational Mapper (ORM) to allow conversion of data between incompatible systems (between models and databases).
It is inbuilt within the Laravel framework.

HTACCESS file: It is the configuration file for Laravel web application framework.

HTTP: Hypertext Transfer Protocol (HTTP) is a request-response protocol which is used to communicate between clients and servers.

HTTP Cookie Manager: It manages cookies containing data on user sessions and login information.

Hypertext Markup Language (HTML): HTML is a standard markup language to create web pages.
Integrated Development Environment (IDE): Software development application with necessary tools and development environment.
JavaScript (JS): JS is a high-level programming language to create interactive web pages.
Management System web application.
JQuery: JQuery is a JavaScript (JS) library that is used to simplify the use of JavaScript programming language.
Laravel: A simple, elegant and well-documented PHP web application development framework (Bautista, 2012).
Model-View-Controller (MVC): MVC is one of the software design patterns. The View consists of the user interface and user experience. The model delivers data to the View. The Controller takes the user's request and loads the appropriate Model and View (Helm, Johnson, Gamma, & Vlissides, 2000).
MySQL: An open-source database management system that stores data in multiple tables instead of using one table to store all data.
Node.JS: It is a server environment which can be used across software platforms.
Open-source: Open-source software can be used, modified or shared for free.
Page Latency: It is the page load time when establishing a connection between the web server and the browser, and vice versa.

PHP: Hypertext Preprocessor (PHP) is a language used for server-side programming for the development of web applications and web pages.
PhpMyAdmin: A graphical interface for the MySQL database server.
Server-side programming: Also known as back-end programming, deals with the logic and functioning of web applications.
Software Design Patterns: Software templates of formalized best practices in the world of web application development.

System with the Laravel framework.

The developer: For this study, a person creating Project Management System web application with Laravel frameworks, and compiling reports based on the experience and literature within the scope of this study.
Windows : An operating system.
Visual Studio: It is an IDE used for the development of Strategic Plan Management System.
Web development framework: It is the implementation of software design pattern that constitutes a set of formalized templates of software features, libraries and best practices used for the development of web applications.
XAMPP: It is a free and open-source web server that can be used across software platforms.

Problem Statement

There are many software design patterns and web application frameworks in the market. It is essential for developers to know the most suitable software design pattern and the web application development framework based on that design pattern. Appropriate framework provides development speed, support features, proper usage, and less expenses. Therefore, as a developer, a good understanding of framework is vital for web development process.

Proposed Solution

The study proposes MVC as a suitable software design pattern for web application development. Laravel is popular web application frameworks based on MVC design pattern. This frameworks is open source, popular, and have vast communities. It is great to build Strategic Plan Managemnt System. The SPMS contains user profile, roles,User Management, and Plans.

Chapter II: Literature Review

Design patterns are reusable solutions to software design problems (Bisson, 2017). Design patterns reuse successful software architectures and provide design alternatives to make a system reusable. They also help document and maintain existing systems (Jailia, Kumar, Agarwal, & Sinha, 2016). Developing generic software applications built on the dynamic web is complex and costly because developers need to understand many platforms, technologies, and architecture. MVC offers the possibility of code reuse and provides separation between the application layers. Platforms, technologies, and dynamic web architectures generate significant cost on a technical level. Developers are required to produce high-quality dynamic web applications within a short time (Komara, Hendradjaya, & Saptawati, 2016). A suitable web application framework reduces much burden from the developer side. Given the popularity of the MVC design pattern, Laravel frameworks, there is a broad consensus over the use of web application frameworks and software design patterns.

Laravel Framework

The Laravel framework relies on Hypertext Preprocessor (PHP) as its scripting (programming) language. Laravel has an expressive syntax that provide solutions for development and facilitate general task in big web projects (Alfat, Triwiyatno, & Isnanto, 2015). Laravel is a clear, simple, elegant, and well-documented framework that focuses on equipping and enabling developers. It helps developers learn, start, and develop quickly (Stauffer, 2016).

The Laravel framework is useful when there are issues of time, security, and complexity.

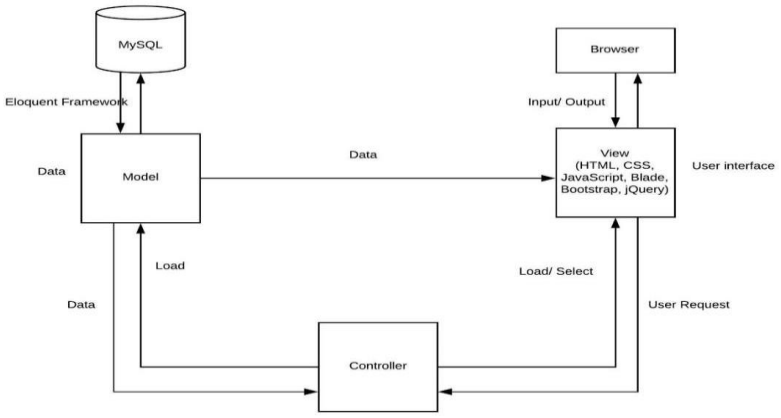


Figure 2. Laravel Architecture Diagram

Strategic Plan Management System. (SPMS)

SPMS is a system where users are added in roles, such as: Administrator, Head Office, Unit Leader and Employee.The Administrator role can add other users to the SPMS for access. However, the Users with privileges other than Administrator are not able to add other users. Once logged in to the SPMS, users can add, update or delete activities from the plans. This system allows users to collaborate on plans or activities assigned to the users.

Feasibility Analysis :considers all important components of a project, such as the economic, technological, and legal aspects, as well as the project timeline, all of which contribute to predicting project success. The system should be user-friendly, costeffective, and simple to maintain and improve. Feasibility study is conducted to test the operational, economical, technical and legal feasibility of the system.

Operational feasibility The system provides a user-friendly interface and is therefore easy to use. Since the system is user friendly the user can operate it and the user can access the system easily to get what he/she wants. The system does not require any programming knowledge to operate, and anyone with a basic understanding of computers may use it. The system is simple to use for students, concerning offices and registerer. Therefore, the system is operationally feasible.

Technical Feasibility One of the most essential criteria for selecting content for digitalization is technical feasibility. The system is technically possible and is used to improve the speed, accuracy and time of technical tasks as well as to minimize technical errors. therefore, it is technical feasible.

Economic feasibility The economy feasibility of this project is concerned on the financial benefits of the system to the organization and concerned office. It reduces the time and efforts takes to finish the processes and maximize productivity of the workers. Because the system is online and computer-assisted, it saves money by reducing the amount of money spent on various tasks in an organization's current system. furthermore, the cost for the system development is eminent when compared with the outcome benefits of the system. Therefore, the system is economically feasible

Schedule feasibility: defined as the probability of a project to be completed within its scheduled time limits, by a planned due date. We have a number of scheduled works on the project with all groups for the sake of simplicity and quick development of the system.

Legal feasibility :The system complies the data processing system's rules and regulations. As a result, the system is legally practicable because it does not clash with any rules or regulations. Therefore, the system is not in violation of any law or regulation. So, it is legally feasible.

Requirements of the proposed system

Functional requirement: are product features that developers must implement to enable the users to achieve their goals and also needs to be clear, simple, and unambiguous. Functional requirements are those that specify what the system must be capable of. There are various functional requirements that the proposed or new system can meet.
For example: The system must allow users to log into their account by entering their email and password.

Non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-functional requirements may be more important than functional requirements. If these conditions are not met, the system may be rendered ineffective. it can be expressed into ways: -

Performance requirements

Time: how long it takes to execute its function at given period of minute and seconds.
Space: amount of storage required in process.

Process requirements

Based on the specified inputs, the system performs services for all inputs through functional components of the system. store data, kept in database permanently, can be retrieving easily and so on.

System Requirements

Table 1 shows the software components needed to develop Strategic Plan Management System web application in the Laravel framework.

Components	Laravel 5 (Dev)
Environment	Windows 10 (Physical)
Object Relational Mapper (ORM)	Eloquent
Template Engine	Blade
License	MIT License
Support and Security	Developed and maintained by Taylor Otwell and the Laravel community on Github
User Interface / User Experience Design	AdminLTE – An open source admin dashboard
Programming Language	PHP
Integrated Development Environment (IDE)	Visual Studio Code
Database Server	MySQL

Chapter III: Design

The design section comprises the description of the MVC design pattern and architecture of a SPMS built with Laravel.

Architecture of SPMS

The high-level flow of information in the SPMS is shown in Figure 3 and 4.

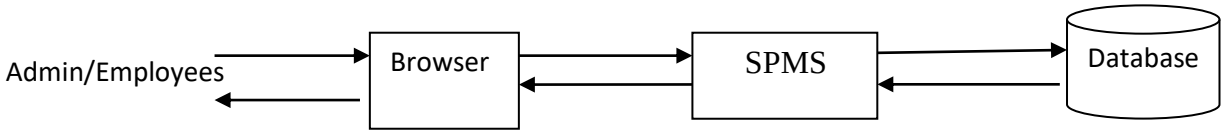


Figure 3. Flow of SPMS

Administrator and Users log into the SPMS with their email and passwords.

They have access according to their roles assigned by the Administrator. For example: Only Administrator has full control of the SPMS and can provide or revoke access of all other users.

Figure 4 shows the detailed architecture of the SPMS built on *Laravel* frameworks.
The web application has insert, update, and delete functionalities.

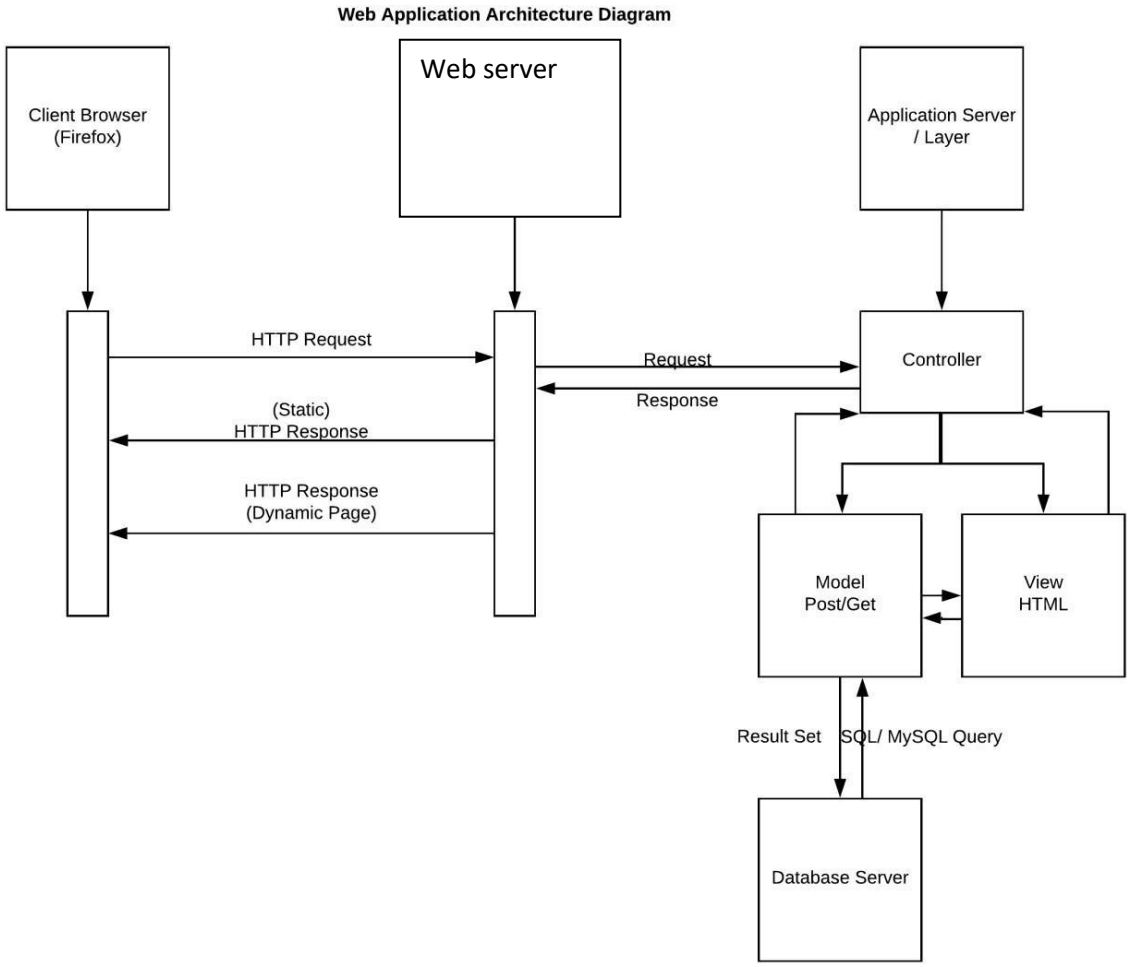


Figure 4. Architecture Diagram of SPMS

As shown in Figure 4, the client browser sends an *HTTP* (*GET*/*POST*) request to the web server. The web server accepts the *HTTP* request and forwards the request to the controller. The controller determines whether it needs to interact with the model or not. If interaction with the model or the database is not needed, it returns a view, composed of *HTML* markup code as *HTTP*, which is a static page. If the controller determines that interaction with the model be needed, it interacts with the model, and then the model interacts with the database server using the *SQL* queries generated by the controller. The data set is generated by the database server, which is interpreted by the controller. Then, the controller responds with a web page.

Chapter IV: Implementation

This section explains the execution of procedures and requirements to develop the SPMS with *Laravel* on *Windows*.

Implementation of SPMS with Laravel Framework in Windows

The developer set up the development environment and planned the development of the application. *Laravel* 5 was downloaded, installed and configured on the development machine using *Composer*. The setup for the web application development involved installation of *XAMPP 7.4.27* which included *PHP 7.4.27*, *Apache HTTP server* 3.3 and *MySQL* database server. The *Composer* and *Node.JS* were also installed in the development machine.

The installation involved configuring the application by adding a *.env* file which contains the basic configuration options for the application. Since *Laravel* is installed with *Composer*, a package dependency resolver for *Laravel*, it is easy to add the few features needed for rapid application development. The *Composer* was used to add an authentication service to the application which allowed the developer to add basic authentication and started to build the application.

Migration service is inbuilt in *Laravel*, so it was enabled in the application to make it easy to migrate the application from the development machine (*Windows*) to the production machine (*Windows*). An *HTACCESS file* along with routing rules were added during the development.

After setup, coding of the application began with adding new controllers, models, views, and resources to the *Laravel* framework. A third-party debugger called *Debugbar* was added to make it easy for the developer to debug the application.

An open-source user interface template called *AdminLTE* was added using *npm*, a default package manager for the *Node.JS*. The developer encountered various problems working with the newly integrated *AdminLTE*. So, the developer had to remove the *npm* dependency on the *AdminLTE* user interface framework.

A stable version of *AdminLTE* was added by the developer without the *npm* dependency. The developer made changes in the *Composer.JSon* file before issuing the *Composer* update command to resolve the dependencies.

The developer encountered several other issues while working on the project, such as a redirection problem while logging out of the application, error in the sidebar of the interface, and a problem in the migration of the permission module. Therefore, the developer had to roll back significant changes to troubleshoot the problems and finally got the application to be stable and run smoothly.

User Interface of SPMS

This section explains the user interface of the SPMS. It consists of the login page, dashboard page,roles page, Plan page. **Login Page**

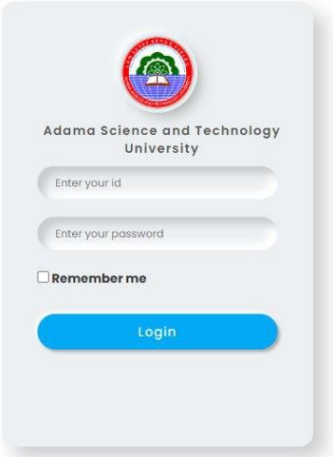


Figure 5 is the login page for the SPMS in *Laravel* on *Windows*.It is the default page that is displayed after launching the application. Users provide their registered email and password to access the SPMS.

Dashboard

Figure 6 is the dashboard page for the SPMS. After logging in, the user can access the dashboard. The left navigation bar provides links to the dashboard itself, *User Management* (*Roles* and *Users*), *Plans* List, Employee,*Change Password,* and *Logout.*

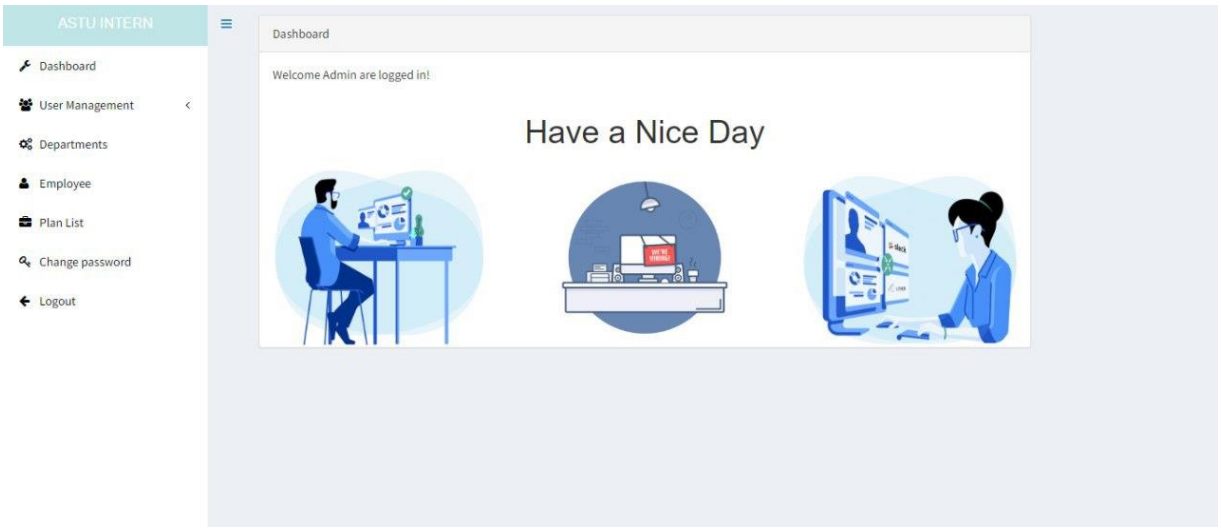


Figure 6. Dashboard Page of SPMS

Roles

The Roles page (Figure 7) lets the *Administrato*r add roles of the users that have access to the SPMS. The roles can be *Administrator* and *User*.

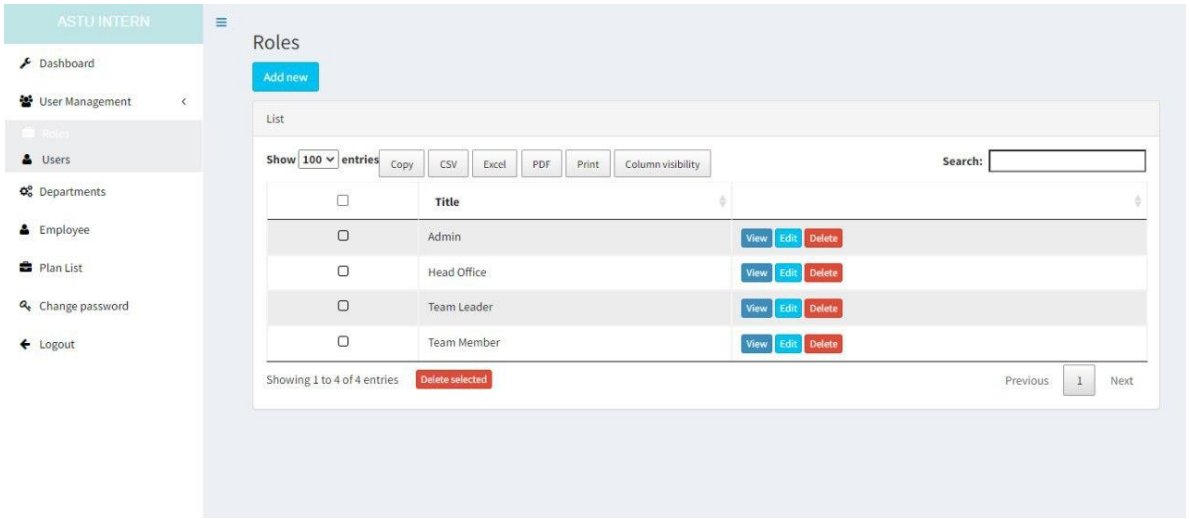


Figure 7. Role Management Page of SPMS

Users

The *Users* page (Figure 8) lists the users who were added to the SPMS. The admin assigns users according to the roles with associated permissions. Only users from the list can get access to the system. The numbers from the *Role* column at the end is the type of access the user has.
For example, the 2nd row with "Computer Science" is as the Role has the "Head Office" privilege.

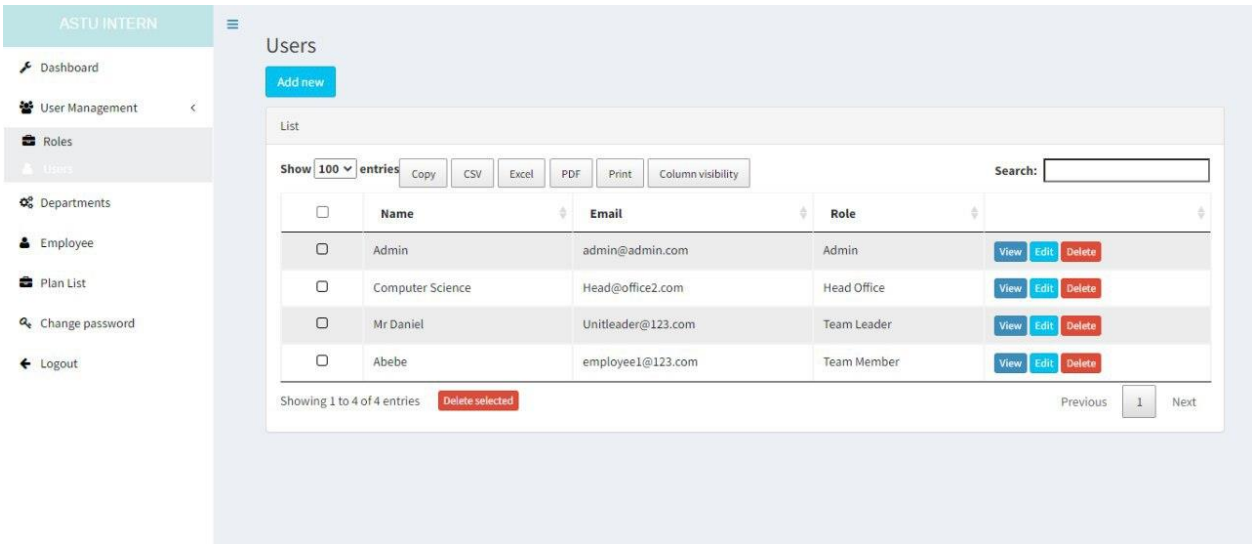


Figure 8. Registered Users in SPMS

The administrator can add new users and assign them roles with *Users* form (Figure 9).

The new user can then access the SPMS with the assigned privilege and the password.

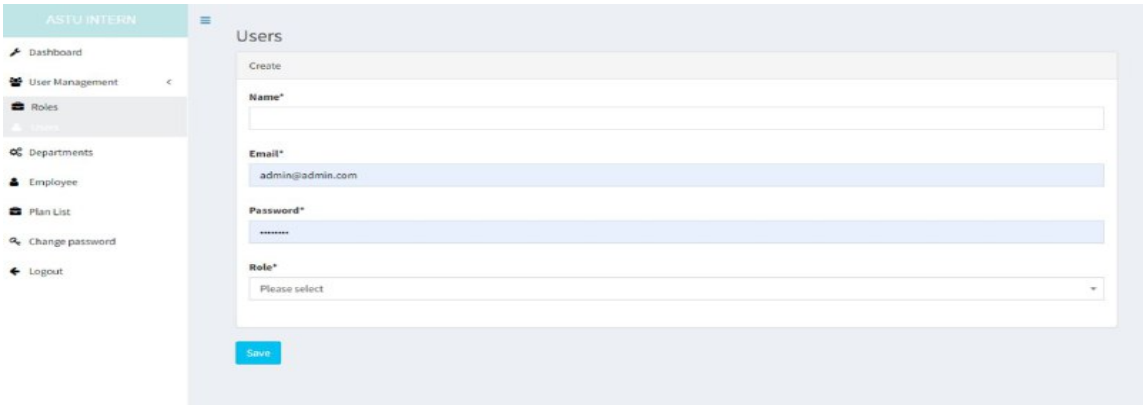


Figure 9. Create User Form in SPMS

The *Plan* page (Figure 10) gives the list of Plans added to the SPMS. It provides the plan name, start date,end date and Actions.

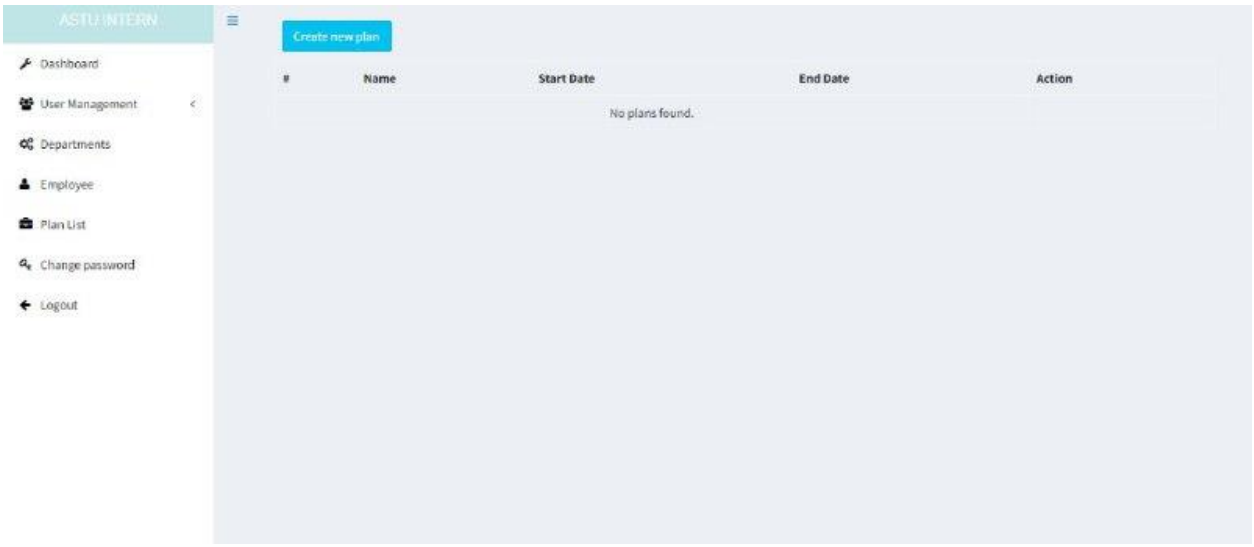
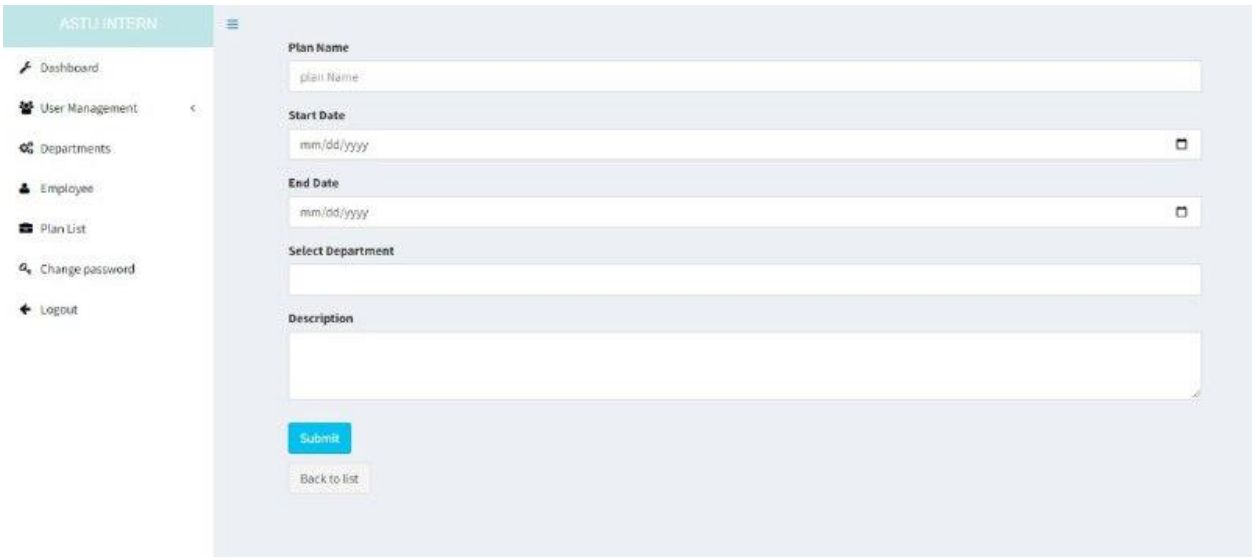


Figure 10.Plan Page of SPMS

The new *Plan* page (Figure 11) lets admin add a new plan in the SPMS.

The Admin can select a name, start date,end date, select department and description for submit.



A: SPMS File Directory in Laravel

Figure 12 shows folders and files of the SPMS in the *Sublime Text Editor* for *Laravel*. Most of the content in the figure is auto-generated. However, it is the developer's responsibility to modify, add and delete files according to the requirements of the web application under development.

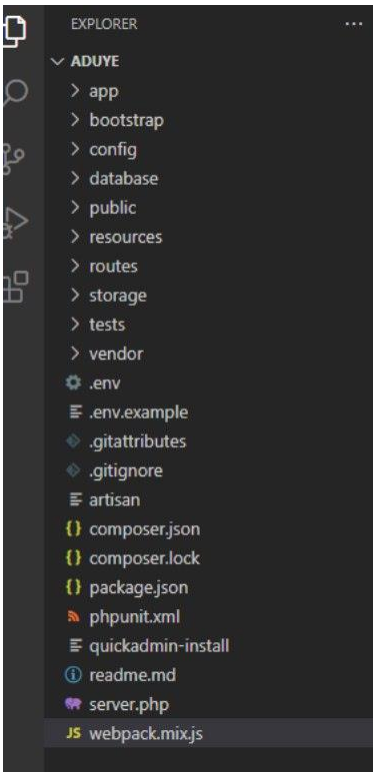


Figure 12. File Structure of the SPMS in the Visual studio code

Following is the list of files added and modified by the developer to the file directory of the SPMS built with *Laravel* framework:

PlansController.php*:* Admin add plans into the SPMS with the help of this file. The plans are shared with other headoffices to view. The plan can be edited and deleted by the Admin.

ActivityController.php: Head office create activities into the SPMS based on plan with the help of this file. The Activities are shared with other Unitleaders to view. The activity can be edited and deleted by the Head office.

***RolesController.php*:** It manages roles in the SPMS. The role can be an administrator or a other privilege user. Administrator has full privilege to add, delete, edit and read, unlike the other user who has limited privileges throughout the SPMS.

***UsersController.php*:** It manages users in the SPMS. It can be accessed only with administrator privilege. It can be used to create a new user, delete the existing user, and update the information of the existing user or list out the available users in the SPMS.

***Controller.php*:** It is the *base controller* class which provides functionalities to newly created controllers.

PlanFormRequest.php: It validates data obtained from the form submitted by the admin. It validates data based on the given parameters. It executes database operation if the validation is successful but denies it if they are not validated.

For example: If a form requires numeric data, but the user supplies special character, then the application will issue a warning that the input data is not valid.

Plan.php: This file inherits its functionalities from the Model class. p*lan* consists of field names such as name, description, head_id which are used to interact with the corresponding field names in the database using *Eloquent ORM* in *Laravel*. These field names are bound in the forms, which in turn carry data from users and interacts with the database. Similarly, *Activity.php, Role.php, and User.php* inherit from the Model class.

***App.php*:** This file provides basic configuration of the application. The properties of the application such as the application name, application environment, debugging settings, application URL, application providers, and aliases reside here.

***Database.php*:** It consists of database connection settings for the application. Such as name and type of the database to connect, and user credentials of the database to connect.

***Migration (folder)*:** It is a feature in *Laravel* that allows creation of tables in the database. For example: During deployment of PMS, it is not necessary to create tables manually in the database, the migration feature in *Laravel* creates those required tables into the database.

***Seed (folder)*:** It allows to insert default data into the tables in the database.

***Public (folder)*:** The public folder consists of web resources such as *CSS*, images, and *JS* files of the application.

***Resources/Views*:** It contains the user interface or presentation part of the application. The controller calls view method and renders layout in these files.

***.env*:** It is a configuration file for *Laravel* which consists of important variable names and their corresponding values.

User characteristics

User to System

To prove their eligibility as a employee, all users must go through the log in page. Users only need to open the login page to enter the necessary information into the form that will eventually appear.

System to User

The system will first check to see if the user requests the form. the system compares each piece of information (particularly names and IDs) provided by the user to the student information in the database to see if they are identical.

General constraints: list or refers to high-level constraints for the software, such as government regulation, hardware limitations, Safety and security considerations, Parallel operation and hard ware platforms. assumption and dependences discuss items that influence the software, but may change. For example, the developers may assume that government regulation are going to change on January 1 because they have always changed on that date, the project will change dramatically.

Use cases

A use case is a list of actions or event steps typically defining the interactions between a role and a system to achieve a goal. Also, it is a description of a set of actions taken by a system to produce an observable result of value to a certain actor. Systems use case describes what a system does, but it does not show how it does it. The system use cases are: -

Actors

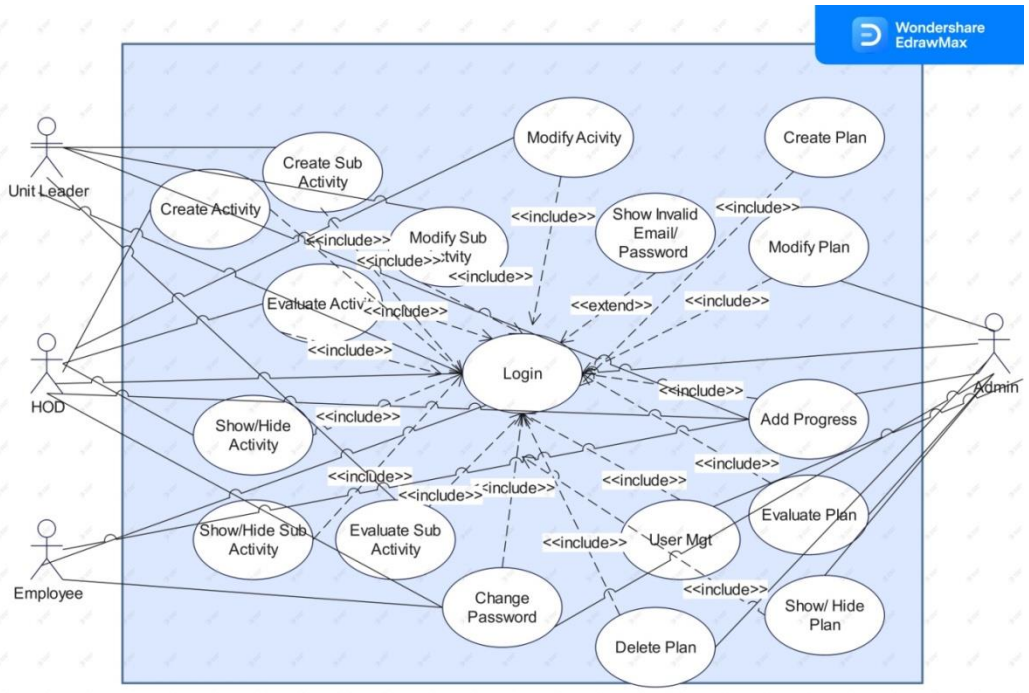
✓ Admin

✓ Head Office(HOD)

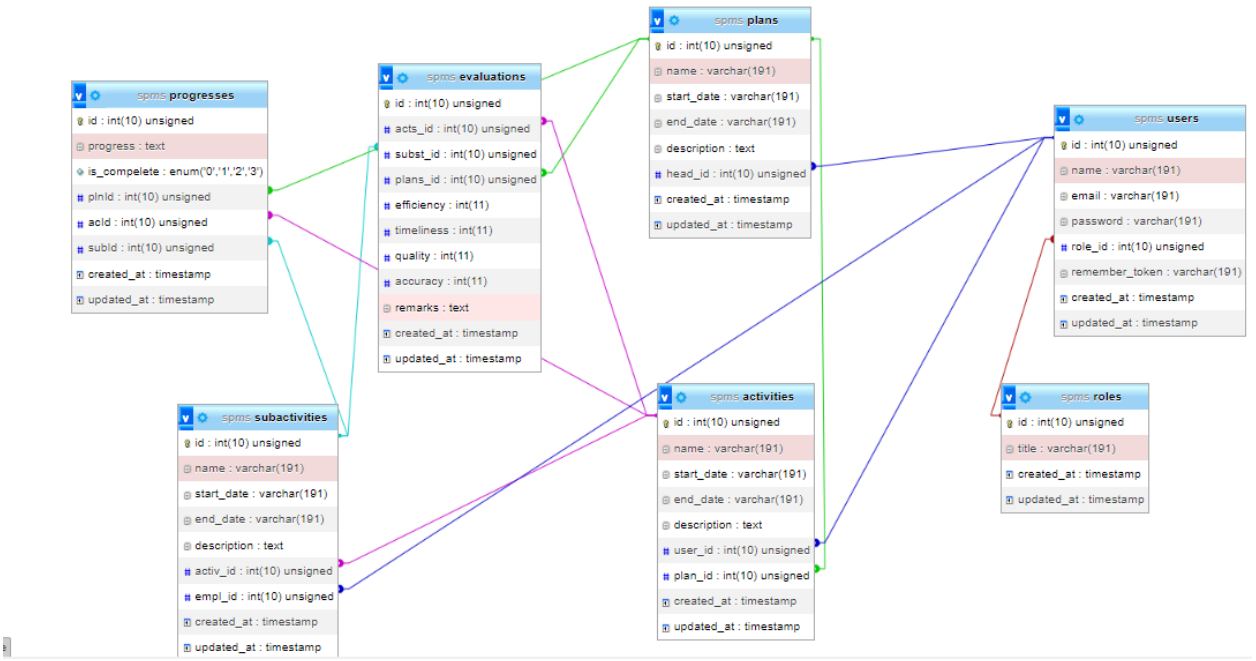
✓ Unit Leader

✓ Employee

Use cases diagram



Object diagram An object diagram describes the instance of a class. It visualizes the particular functionality of a system. The goal of this topic is to see how well those component elements function together to achieve their goals. UML modelling will be used to represent the suggested system, which is a language for visualizing, describing, building, and documenting the artefacts of software development.



B: Sequence Diagram of SPMS in Laravel

Figure 18 shows a user sending a request to display a form via an interface. SPMS accepts the request and responds with a login form. The user fills up the login details and submits the form. The application validates the user details, and if the validation is successful, SPMS redirects the use to the dashboard page. If the validation is unsuccessful, the application returns the login page with validation errors.

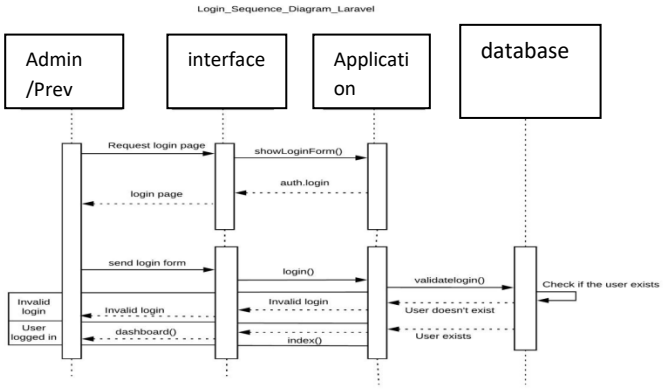


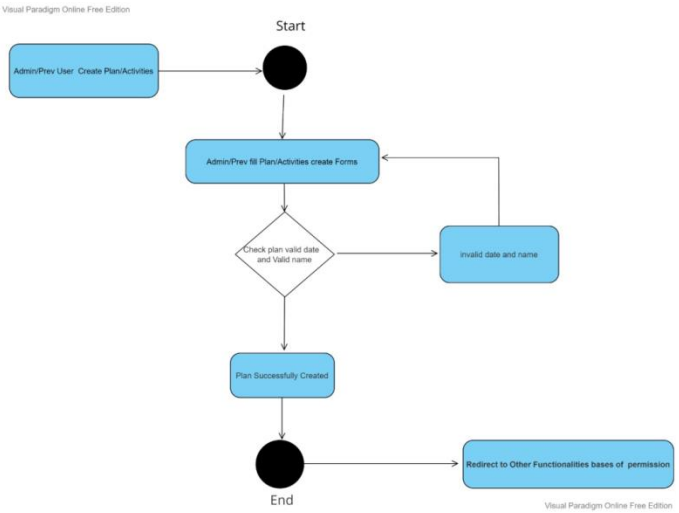
Figure 13. Admin/User Login Sequence Diagram of SPMS in *Laravel*

Activity diagram

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join.

Figure 13. Admin/ Prev User Creating Plan/Activities Activity Diagram of SPMS in *Laravel*

After intern home page of previledge page



Figure, that shows Activity diagram for creating new plan/activities of the system

CONCLUSION

The result of the plan is described from the perspective of the aim and scope set in the beginning of the thesis. The ideas for the future web-based project management system are also described here.

The aim of the project was to make a complete, fully working web based project management system for the company. Requirements from the company has been gathered and taken into account. In web based project management system there has been used an already implemented TRAC system to improve company‟s everyday use and to increase performance, productivity and efficiency. As a good project management system it has a possibility to upload, download and delete files and uniformly gives change for developers to be in constant contact with the customer requirements and expectations for the project. User management tool in web based project management system is a good appliance for keeping eye on the project and for giving rights to different users by system administrator in company. This all makes a complete and good communication system inside company, all data and material will be accessible from one place, to facilitate the solution of a project and contact communication with a client. Finally, the whole system has been tested to ensure that everything functions correctly before the system processes actual data and produces information that people will relay on. The features that were implemented are listed below.

General principles of system, implemented:

1. Users management
   - ✓ Adding new users
   - ✓ Adding reliable data to user
   - ✓ Ability to add different rights to users (admin)
   - ✓ Ability to change password
   - ✓ Ability to delete users
   - ✓ Ability to lock users (user don‟t have possibility to log-in to system)

- ✓ Ability to modify data and further on to change rights

2.Plan management

- ✓ Adding new plans to system
- ✓ Adding projects logo to Trac system
- ✓ Define a admin for a specific plan
- ✓ Adding needed data to plan
- ✓ Updating plan
- ✓ Deleting updated plan

2.Activity management

- ✓ Adding new Activitys to system
- ✓ Adding projects logo to Trac system
- ✓ Define a admin for a specific Activitys
- ✓ Adding needed data to Activitys
- ✓ Updating Activitys
- ✓ Deleting updated Activitys

4 Sub Activitys management

- ✓ Adding new Sub Activitys to system
- ✓ Adding projects logo to Trac system
- ✓ Define a admin for a specific Sub Activitys
- ✓ Adding needed data to Sub Activitys
- ✓ Updating Sub Activitys
- ✓ Deleting updated Sub Activitys

5.Plan/ Activitys /Sub Activitys based views (Every plan has its own Trac project)

6.Security login added to system (authentication of a user)

To conclude, web based Startegic plan management system is an improved TRAC system and completed as one fully functioning system that corresponds to the Organization needs and helps to produce a good quality web application plans for the clients. The result of the project responded to the customer‟s expectations. The company was satisfied with the features implemented and their reliability and robustness. Through the thesis and

development process I gained quite good experience, of an overall structure of different systems and the basic concept of the system as whole. It was quite challenging to improve already made system as TRAC, while adding there new features and to put TRAC to work with new functions. New ideas of what more to improve or how to improve the system and what kind of new features to add, come up through the development of thesis. For example there has been an idea, to make search functionality to project, which will help users to search projects by date or by creator of a project. During the process of implementation, wonderful ideas have been got and hopefully in a near future, there is a possibility and time allocated to improve the system.

References

Alfat, L., Triwiyatno, A., & Isnanto, R. R. (2015). *Sentinel web: Implementation of Laravel framework in web-based temperature and humidity monitoring syste*m. 2015 Second International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), 46-51.

Bautista, N. (2012*). Building web applications from scratch with laravel*. Retrieved from HTTPs://code.tutsplus.com/tutorials/building-web-applications-from-scratch-with-Laravel--net-25517

Bisson, S. (2017). *Net framework or .Net core? When to use which*. Retrieved from HTTPs://www.infoworld.com/article/3180478/development-tools/net-framework-or-netcore-when-to-use-which.HTML

Helm, R., Johnson, R. E., Gamma, E., & Vlissides, J. (2000). *Design patterns elements of reusable object-oriented software*. Charlesbourg, Québec: Braille Jymico Inc.

Jailia, M., Kumar, A., Agarwal, M., & Sinha, I. (2016). *Behavior of MVC (Model View Controller) based web application developed in PHP and .NET framework*. 2016 International Conference on ICT in Business Industry and Government (ICTBIG), 1-5.

Komara, H., Hendradjaya, B., & Saptawati, G. P. (2016). *Dynamic generic web pattern formultiplatformm*. 2016 International Conference on Data and Software Engineering (ICoDSE), 1-5.

Stauffer, M. (2016). *Laravel: Up and running: A framework for building modern PHP Apps* (6th ed.). Sebastopol, CA: O'Reilly Media.