Bachelor’s thesis

Bachelor Degree in Information Technology

2019

Abhinay Chaudhary

OCTOBERCMS: A CONTENT MANAGEMENT SYSTEM FOR WEB DEVELOPMENT

TURKU AMK

TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Bachelor Degree in Information Technology

2019 | 43 pages

Abhinay Chaudhary

OCTOBER CMS: A CONTENT MANAGEMENT SYSTEM FOR WEB DEVELOPMENT

Web Developers often choose content management systems when building websites because they offer built-in features which accelerate and simplify web site deployment. The purpose of the thesis was to become familiar with content management systems and their features. The thesis describes the properties of October Content Management System (CMS), their strengths as well as weaknesses, and the Laravel Framework on which October CMS is built. The use and operation of the October CMS platform are further illustrated with a practical example of website development which needed the implementation of other contemporary web technologies.

KEYWORDS:

October CMS, CMS, Laravel Framework, CMS platform

CONTENTS

FIGURES

CODE SNIPPETS

TABLES

LIST OF ABBREVIATIONS (OR) SYMBOLS

CLI = Command-line Interface
CMS = Content Management System
cURL = Client URL Library
GD Library = Graphics Drawing Library
MVC = Model-View-Controller
OpenSSL = Open Secure Sockets Layer
PDO = PHP Data Objects
PHP = Hypertxt Preprocessor
RAD = Rapid Application Development
URL = Uniform Resource Locator

1 INTRODUCTION

Today, technology offers us the opportunities that we did not even imagine two decades ago, and we did not know that it would change our way of life so much. Since its initiation in the 1990s, the World Wide Web has undergone rapid development and progress. In their beginning, the websites were simple and had only a few subpages, but their content did not change much (Laminack, 2001). With the increased complexity of the sites, problems began to appear, as content editing became more time-consuming, and the content started to change more quickly. When we wanted to edit the content of the websites, it was first necessary to obtain the desired files from the server, edit them and then reload to the server so that we could replace older files with new ones. With the development of Content Management Systems (CMS), the editing of websites content has radically changed, since modern CMS systems are very functional and can significantly help us in the development and management of websites. They allow us to edit the content of the webpages within the user interface in browsers, which means that we avoid the delayed procedures and possible errors when editing individual web pages and uploading files to servers. For better content management and other features such as user control and plug-ins, they have become a popular choices for ordinary users who, with a variety of components within the user interface can create their own websites in a few steps, as well as for business and organizations, which through their specific CMS systems manage their business and business processes and monitor the course of work and organization. (Smooke, 2017)
Developers often have difficulty in choosing the right platform for creating and developing websites. Nowadays, there is a great variety of CMS platforms on the market. Developers strive to ensure that the systems are safe, reliable and that they enable high-quality website development. Due to such factors, we decided to learn about and explore the platform of the new generation of the October CMS which is built on the most popular PHP framework ,Laravel, by building a simple website.
The purpose and the goal of the thesis is to get to know the October CMS and its features by developing a website for IDNepal. First, we will present and describe the properties of CMS systems, advantages and disadvantages of their use. Next, we will present the Laravel framework and MVC architecture, and finally describe the platform, its properties, the construction of the templates, and the tool for the fast-visual development of plug-ins. Based on the development of the website, we will show the functioning of the platform and its suggestions. We will also explain how the determination of the route and other important features that we will get to know during the creation of the website will take place. Within the framework of the October CMS, we will also describe the mechanism for managing the "Twig" templates, which October CMS uses to build its proposals. We will create a web page from the beginning with the help of the widgets that build modern websites.

2 CONTENT MANAGEMENT SYSTEMS

Content Management System is a software package that enables us to automate particular operations in the system and thus ensure efficient content management. Depending upon the nature of CMS platforms and their constraints, every user can manage the variety of content types and set the specific implementation process. CMS platforms allow users to create, modify, delete or publish content on websites with the customizable user administration to ensure only the authorized access and the room to expand according to the need.(Barker, 2016)
One of the most considerable advantages offered to us by CMS platforms is that the user does not have to know more about technologies, websites designs and other processes that take place within the systems. An inexperienced user who wants to create a simple blog makes it easy with the help of already made templates, and then with some simple steps changes the look and functioning of the site. Of course, the use of CMS is not limited to simple websites, such as blogs and personal websites, because these systems are also used by business, corporations, government organizations, online newspapers and prestigious educational institutions. However, for the creation of sites, good knowledge of the system itself is required, for this reason, a programmer or a developer who has expertise with the system and its operations is needed. Despite technical skills like programming or designing, CMS helps every user to create and manipulate various contents that can be published on the website and build a comparable system alone or through a variety of frameworks.. (Cms.co.uk, n.d.)

3 OCTOBER CMS

October CMS is a self-hosted open source CMS platform build in PHP based on the Laravel framework. Alexey Bobkov and Samuel George developed this CMS for the rapid development of the web in a simple way (About - October CMS). On 15 May 2014, the first beta testing version was released. (Rizwan, 2018)

3.1 Laravel Framework

To understand October CMS let us briefly discuss the Laravel framework. Laravel is an open-source based on MVC architecture implemented on most popular scripting language PHP. It was created in 2011, by Taylor Otwell to overcome the weaknesses of available PHP frameworks and to minimize the initial cost of web development as well as to improve the quality of code, by defining standards of design practices. From the first release, Laravel supported built-in user authentication and authorization which lack in other available PHP-powered frameworks. With the development of different versions, Laravel provides different useful features like routing, modularity, file management, session management, view templating engine, caching, validation, database access and many more features that help in rapid web development (Nilanchala, 2017). Among all of the features, Laravel acquired one of the key features known as MVC structure in its early version (2nd version) which provides fully functional systems from front-end to backend. Later, MVC will be briefly described in this chapter. Due to the advancement in the functionality of Laravel like modularity, routing, caching, application testing, sessions management, views, query building, validation, pagination and many more, every developer facilitates faster designing and development of the application including October CMS. (Surguy 2013)

3.2 MVC Architecture Pattern

Laravel entirely follows the MVC (acronym for "Model-View-Controller") architecture pattern for application development (Ighodaro, 2018). Since Laravel is an MVC framework from its second version, October keeps on the following principle of MVC pattern and divides the modularity of application in three different parts (Otwell, n.d.):

3.2.1 Model

Model is the representation of business logic and data structure of its application. Depending upon the model structure, data are stored in the database, which can be obtained using the interface, processed and displayed in the view and also provides reverse logic to store data into the database. (www.tutorialspoint.com, n.d.)

3.2.2 View

View defines the visual representation of data obtained using the model to the user. Therefore, it can be proclaimed that views represent the user interface of the web application and provide access to modify data. (Tutorialsteacher.com, n.d.)

3.2.3 Controller

Controller handles the request coming from the user and process the business logic according to the request, deploys and renders the data obtained from model to the views and vice-versa. (www.tutorialspoint.com, n.d.)

Figure 1 shows the business logic process of MVC in Laravel Framework

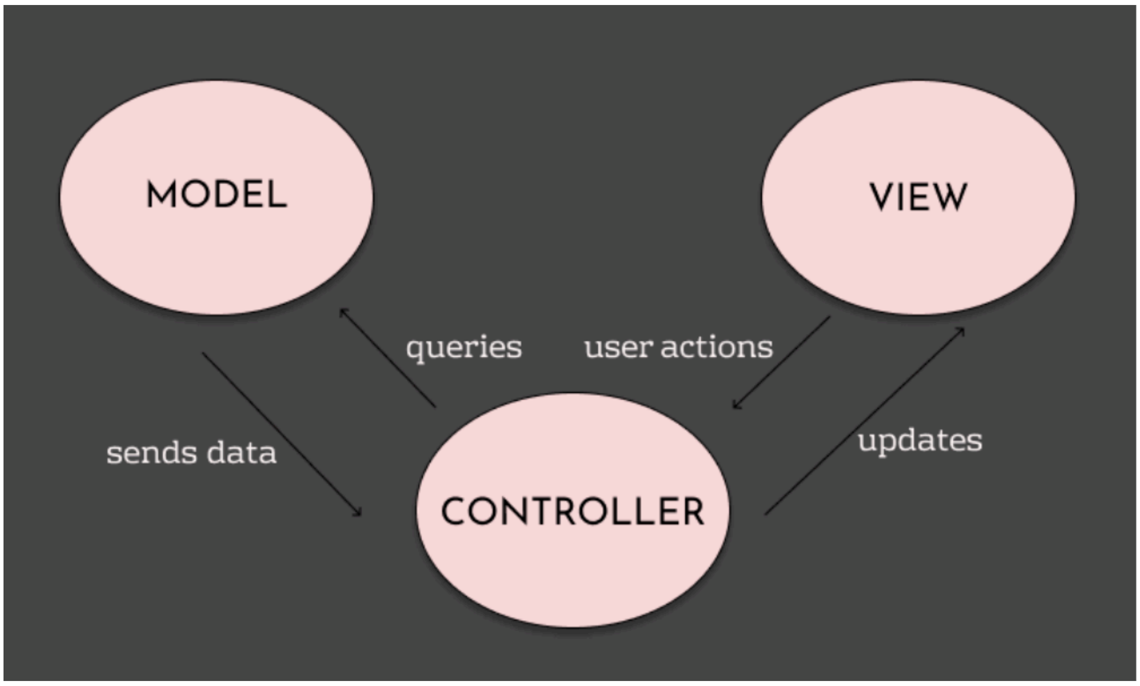


Figure 1. MVC structure (Ighodaro, 2018).

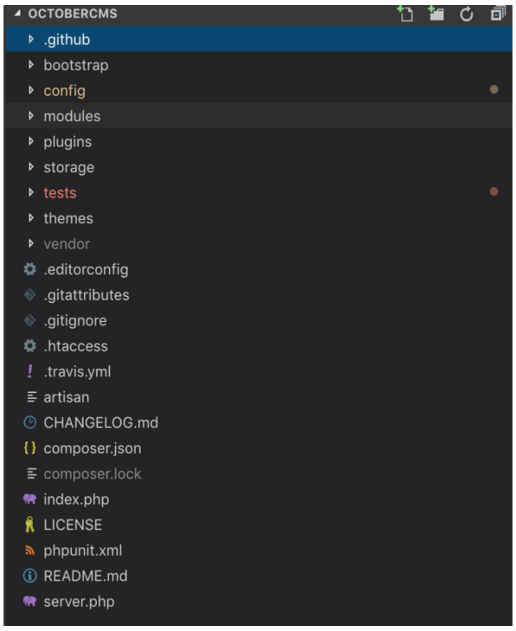
Depending upon the description of individual components and their properties, it can be stated that the development of web applications in the MVC architecture is straightforward to follow and understand.

3.3 October Content Management System (October CMS)

As October CMS is built on the top of the Laravel framework, it handles and manages both simple and complex web applications. In addition, it has a built-in content management system that allows controlling backend of the application, user authentication as well as authorization, plugins and its contents. October CMS also has an intuitive and user-friendly interface that enables to manage the content of the application and entire system. It follows a special approach called "Developer Centric Platform" and focuses on developers, meaning it follows the principle: "Developers develop websites, and the role of the subscribers is to communicate their business requirements and manage the site and its content". For this reason, it is necessary to have know-how at least basic knowledge about web technologies skills, such as HTML, CSS, and PHP to edit, modify, and delete contents of the applications/ websites built.(CMS, 2015)(archybold.com, 2015)
Due to its unique approach, October CMS discourages the use of the CMS by users who do not have the technical knowledge and experience to be able to undertake the development of web application or web sites. The reasons mentioned above put October CMS in a special category of rapid application development. October CMS also has its plugin which enhances insignificant development way, which will be discussed later.

3.4 Directory Structure

Directory structure of October CMS is a hierarchical tree structure for the file system presented to the user after installation of a project, as shown in Figure 2.



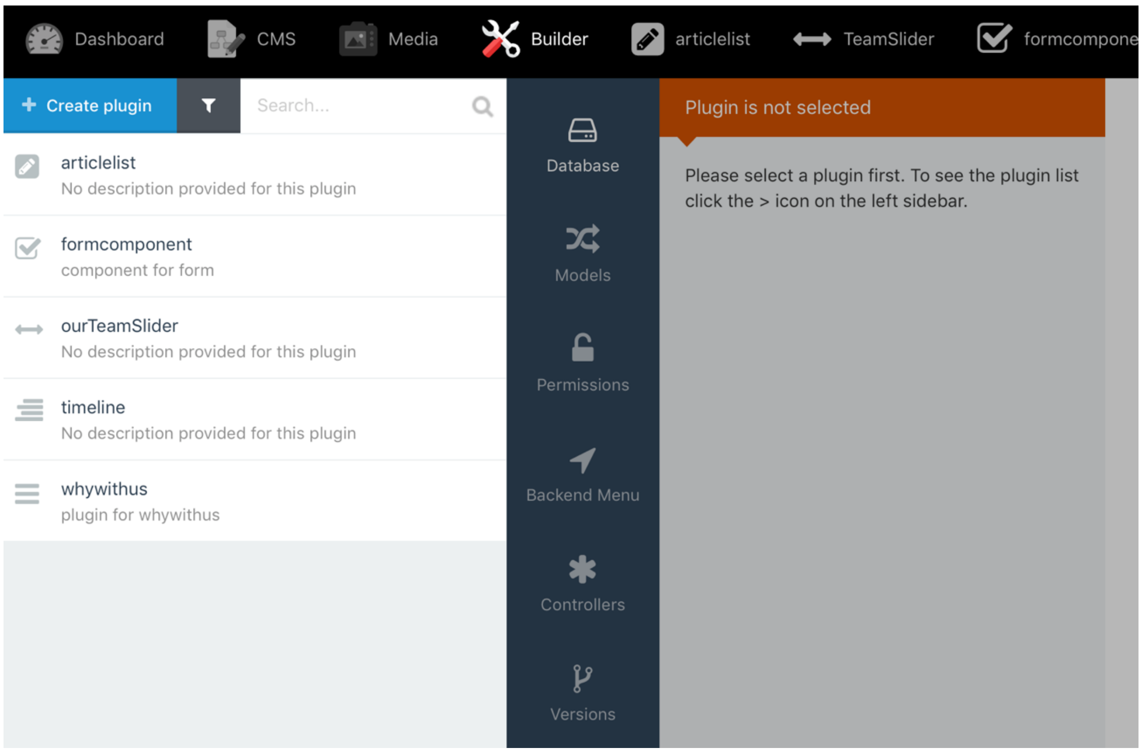
```
OCTOBERCMS
  .github
  bootstrap
  config
  modules
  plugins
  storage
  tests
  themes
  vendor
  .editorconfig
  .gitattributes
  .gitignore
  .htaccess
  .travis.yml
  artisan
  CHANGELOG.md
  composer.json
  composer.lock
  index.php
  LICENSE
  phpunit.xml
  README.md
  server.php
```

Figure 2. Example of Directory Structure of October CMS after installation.

3.5 Plugin Development

According to Wikipedia - “*In computing, a plug-in(or plugin, add-in, addin, add-on or addon) is a software component that adds a specific feature to an existing computer program. When a program supports plug-ins, it enables customization.”* (En.wikipedia.org, n.d.)

From the early development of the October CMS, the developer focuses on minimizing development cost by making the process rapid without hindering developing experiences. To enhance the rapid development process, the developer of October CMS follows the concept of rapid visual development of application solutions called RAD (Rapid Application Development). The purpose of RAD is to facilitate and reduce the time consumed in development throughout the automation of some hectic processes, such as writing code to add more functionality to the application. In 2016, developers created "Builder Plugin" which provides visual development of plugins in the backend of the application as shown in Figure 3(Octobercms.com. n.d.b). The plugins created due to help of “Builder plugin” still have complete control over the plugin and can be updated, modified, improved in case of complications without any pain in the head.



articlelist
No description provided for this plugin

formcomponent
component for form

ourTeamSlider
No description provided for this plugin

timeline
No description provided for this plugin

whywithus
plugin for whywithus

Plugin is not selected

Please select a plugin first. To see the plugin list click the > icon on the left sidebar.

Figure 3. Builder plugin interface.

The Builder Plugin has several tools and functions to create plugins as needed and can be customized as required as shown in Figure 3.
It enables some functionalities such as (Octobercms.com. n.d.b):

- Initiation of the plugin – Creates plugins as needed by the application
- Database – Creates a database for data storage

- Model – Creates models and classes, and forms for the editor in the background of the system to work with plugin data
- Controllers – Creates and manages controllers and configuration for the backend-operations
- Permissions – Manages user permission for the plugin
- Components – Creates components that allow displaying plugin data on the front-end of the application
- Version – Shows version of the application to keep records of changes and modifications
- Localization – Manages locale language for the plugin and by default adapts system language as scripted

3.5.1 Directory Structure of Plugin

Every file and component has a particular directory for its proper functionality. Every plugin has its tree structure to store data, and they generate their workspace; which provides a protocol to access the resources within their directory as shown in Figure 4.

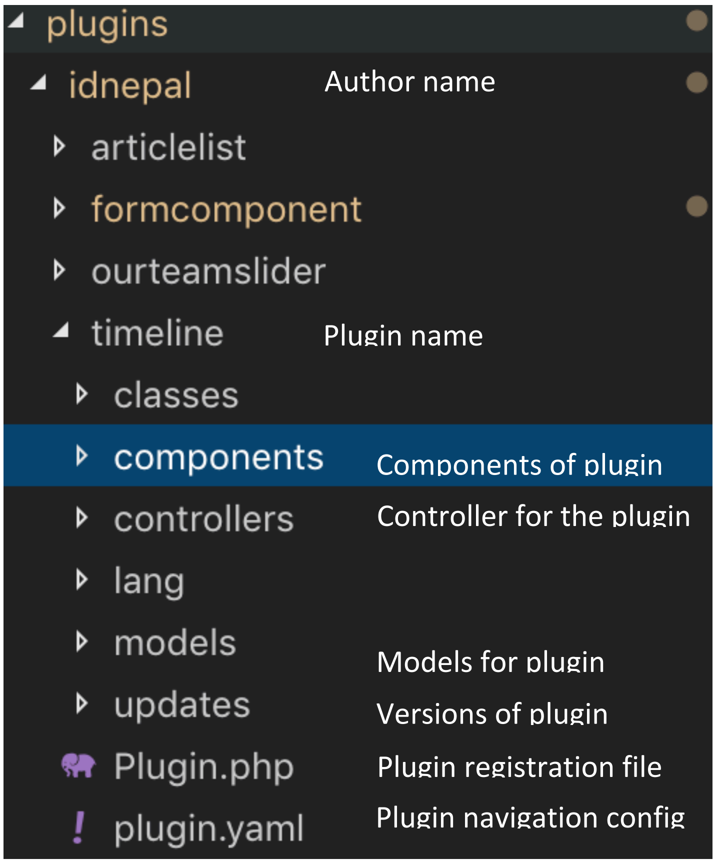


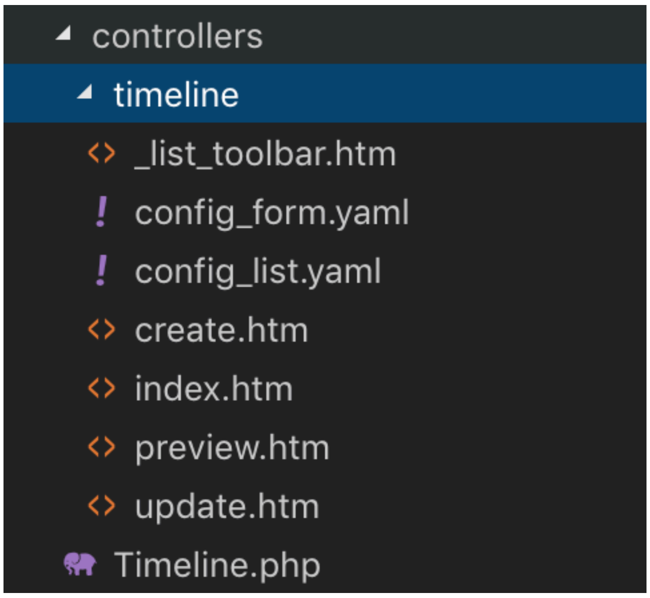
Figure 4. Example of the directory structure of Plugin created using Builder plugin of the October CMS.

Plugins exist in inside the plugins directory of the application as shown in the figure above Figure4.
Let us discuss briefly about the subdirectories of the plugin directory
Plugin namespaces play a vital role in publishing plugins on the October CMS Marketplace. While registering plugin, the author code will be asked as a root namespace for the plugins in the marketplace. The author of the code will be asked once while registering plugin in the market which consists of the first name and last name of the author following upper camel casing: for example JohnDoe.(CMS, n.d.)

3.5.2 Subdirectories of the Plugin directory

- Controllers – Controllers in the plugin administers backend-pages and implements different skins like forms and lists. Each controller consists of a PHP script which exists inside the /controller subdirectory of Plugin directory. The controller view directory name must match the controller class name written in lowercase and also contains config files for the controller. (Octobercms.com, n.d.e)

  Figure 5 is an example of a controller directory inside the plugin directory

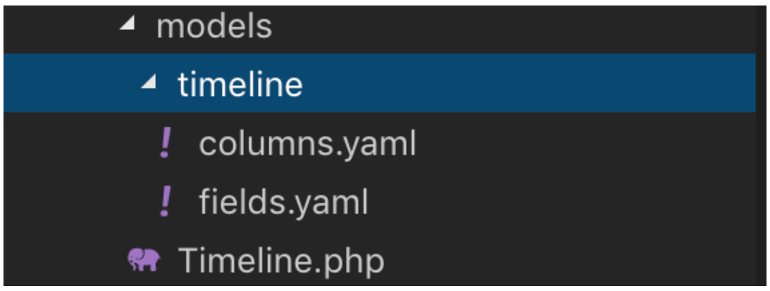


```
controllers
  timeline
    _list_toolbar.htm
    config_form.yaml
    config_list.yaml
    create.htm
    index.htm
    preview.htm
    update.htm
  Timeline.php
```

Figure 5. Example of Files inside the controller directory of the plugin.

- Models – Models are created to interact with the database for record implementation, based on Eloquent by Laravel in a simple way. The individual database table has the respective model to communicate with data inside the database created. The model exists inside of the plugin directory as a subdirectory. The Model directory name must match the model class written in lowercases and contains model's list column and form field definitions inside directory as configuration files for the model(Octobercms.com, n.d.g).

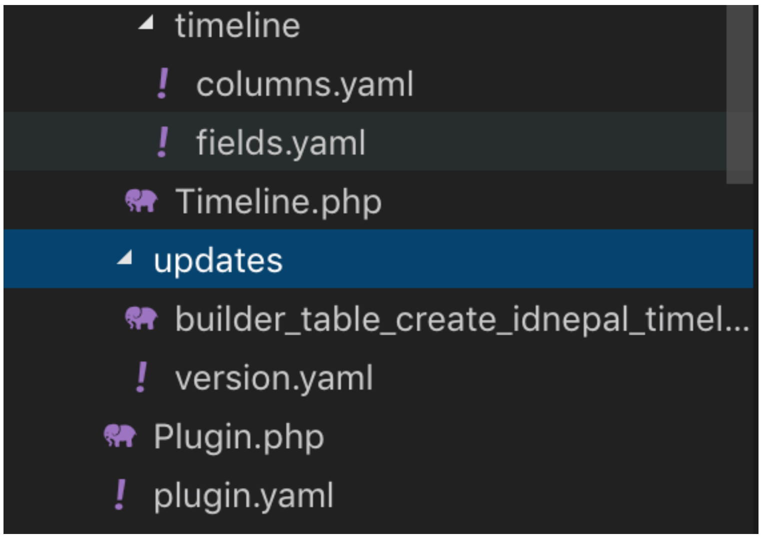
Figure 6 shows an example of subdirectory structure of the model inside a plugin directory.



```
◢ models
  ◢ timeline
      ! columns.yaml
      ! fields.yaml
    Timeline.php
```

Figure 6. Structure of the model as a subdirectory of the plugin.

- Updates – Updates directory of a plugin keep records of any changes occurred or progresses made in the code and perform migration and seeding of the data inside the database of the application after a successful update. The Update process comes to effect when administrator login to the backend of the application or when the update feature is triggered inside backend or php artisan october:up is called in the console command of application. All the changes are stored in the file version.yaml file inside /updates subdirectory of the plugin. (CMS, n.d.)
  Figure 7 shows a structure example of updates directory as a subdirectory of a plugin.



```
    ◢ timeline
        ! columns.yaml
        ! fields.yaml
      Timeline.php
  ◢ updates
      builder_table_create_idnepal_timel...
    ! version.yaml
  Plugin.php
  ! plugin.yaml
```

Figure 7. Structure of update subdirectory with the migration of model.

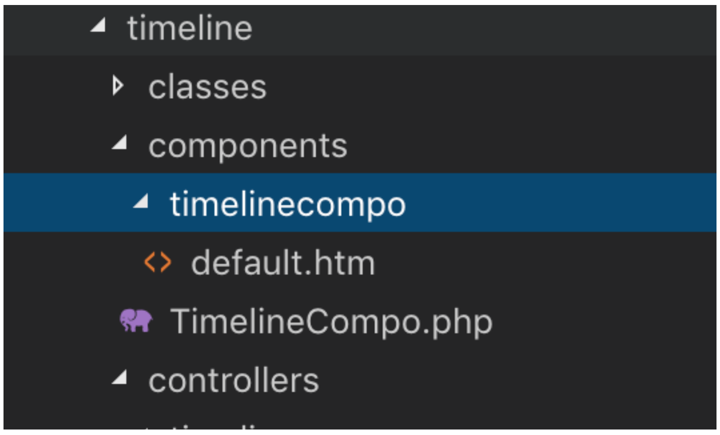
- Plugin Registration file – The Plugin.php file inside the plugin directory called as a "Plugin Registration file", which initializes the script declaration for core functions and information of the plugin. The Registration file provides information about the plugin, name of the plugin, author of the plugin, and the supported methods for extending the CMS(CMS, n.d.). Table 1 shows methods that are supported in the registration file of the plugin.

Table 1. Supported methods in the registration file of the plugin. (CMS, n.d.).

Method	Description
pluginDetails()	returns information about the plugin
register()	register method, called when the plugin is first registered
boot()	boot method called right before the request route
registerMarkupTags()	registers additional markup tags that can be used in the CMS
registerComponents()	registers any front-end components used by the plugin
registerNavigation()	registers backend navigation menu items for the plugin
registerPermissions()	register any backend permissions for the plugin
registerSettings()	registers any backend configuration links used by the plugin
registerFormWidgets()	registers any backend form widgets supplied by the plugin
registerReportWidgets()	registers any backend report widgets, including the dashboard widgets
registerListColumnTypes()	registers any list column, column types supplied by the plugin
registerMailLayouts()	registers any mail view layouts supplied by the plugin
registerMailTemplates()	registers any view templates supplied by the plugin
registerMailPartials()	registers any mail view partials supplied by the plugin
registerSchedule()	registers scheduled tasks that are executed on a regular basis

Components **–** The Components file resides inside the /components directory of the plugin directory, which contains the front-end files for the plugins for the development of the application. Components must be registered in plugin registration class with registerComponents() method.(Octobercms.com. n.d.c)

Figure 8 shows the structure directory for the components inside the plugin directory.



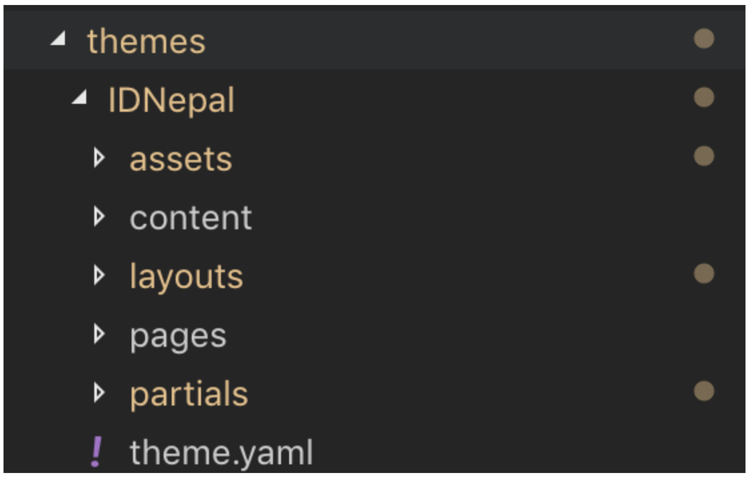
```
◢ timeline
  ▷ classes
  ◢ components
    ◢ timelinecompo
      default.htm
    TimelineCompo.php
  ◢ controllers
```

Figure 8. Structure of component inside the plugin directory.

3.6 Components for Web Site Development

This section focuses primarily on the components that are required for the "Front-end" development of the October CMS powered websites, which means that we will briefly describe the individual components that help to build the front-end. Front-end allowsthe user to interact with the websites.

Figure 9 shows the exampleof directory structure of themes.



```
◢ themes
  ◢ IDNepal
    ▷ assets
    ▷ content
    ▷ layouts
    ▷ pages
    ▷ partials
    theme.yaml
```

Figure 9. The directory structure of web development component front-end.

- assets – contains necessary elements that help us to create our website. We can include fonts, images, JavaScript, CSS, SASS, Bootstrap, LESS, and other more that are needed for the development of the website. In addition, we also can modify the structure if needed.
- content – contains the static pages that do not need change and supports HTML simple markdown.

- layouts – the layouts folder captures the file by which we determine the basic layouts and the format of our website. The file inside layouts represents the main structure and HTML elements. In the element head and body, we can also include all the necessary styles and scripts that ensure the correct display and operations of the websites.
- pages – the pages folders cover the vast majority of the contents of our websites since each of the sub-pages has its own file containing content and elements that are displayed only within the subpages.
- partials – the partial folder contains macro pages that can be used in the website repeatedly in the process of the development.
- theme.yaml – this is a file that can be created manually or within the October CMS user interface. It represents the definition of the theme that we want to create. Tn the event that this does not exist, the system does not detect our newly created theme, which means we can not activate the theme for the page of the website.

Although components are in the plugin folder, they can be called in the individual pages or elements if we want to extend the functions of our web app or website.

3.6.1 Web Technologies

This section briefly describes the web technologies that we used to develop a plugin for OctoberCMS. To communicate between the front-end, backend, and OctoberCMS Platform Twig Template management mechanism is used, which by default mechanism of this platform for intercommunication. Some of the Ttechnologies that will be used in the development are listed below:

- Twig - Twig is an open source mechanism for managing templates, inspired by text-based motion management mechanism such as Jinja, Django, Smarty, and allows us to display PHP code in templates. This means that we can include complete or partial templates in our plugin pages using variables, loops, links, functions, filters, truncate, split, and many other elements with increased security.
  In this process of development, we use two syntactic delimitations which are described below:
  - {{…}} – It will be used to display variables and components of the plugins
  - {%…….%} – It will be used to implement logic terms such as conditionals statements, loops, and functions that will be applied within the twig mechanism.

  The most significant advantage of using the twig mechanism is to speed up and make more accessible to create templates since it allows us to write a simple PHP code that is more complicated and optimized compared to the regular PHP code and at the same time more suitable for use in web templates. To maintain the principles of the PHP programming language and add features that are useful for web development. (CMS, n.d.)

- HTML- HTML stands for Hypertext Markup Language. It is used to build the basic foundation and structure of web application using HTML elements and tags. (W3schools.com, n.d.c)

Code Snippet 1. For example, the basic foundation of the website.

```
<html>
<head>
<title>Title of the Website</title>
</head>
<body>
<h1>Welcome To the world of Web Development. </h1>
</body>
</html>
```

- CSS – Cascading Style Sheets (CSS) is style language to determines the style and defines the way of displaying elements in the page. (W3schools.com, n.d.b)
- jQuery – It is a JavaScript scripting library, which helps to develop interactive web pages with visual effects and modifies the properties of elements with less code.(CodinGame, n.d.)
- Bootstrap 4 – It is the most popular CSS framework used in web development for the quick and responsive web application. This front end-framework contains HTML and CSS based design templates for different types of forms, buttons, tables, navigation menus, image carousel, and many more. (W3schools.com, n.d.a)

4 ENVIRONMENT SETUP FOR PLUGIN DEVELOPMENT

4.1 Minimum Requirements

As we know, for every application or system there are specific requirements to be fulfilled to run smoothly so for October CMS also have some requirements for install as illustrated in the official website of October CMS.(Octobercms.com, n.d.f)

- PHP version 7.0 or higher
- PDO PHP Extension
- cURL PHP Extension
- OpenSSL PHP Extension
- Mbstring PHP Library
- ZipArchieve PHP Library
- GD PHP Library

To meet all these requirements, this project will use MAMP as a local server environment. There is a lot different local server environment like XAMP, WAMP is popular in the market and can be installed manually in the pc itself but have to install everything individually. The purpose of installing the platform is to use the Apache Server and MySQL database within the MAMP server and get access through user interface called phpMyAdmin. We created a database using the phpMyAdmin interface, which the system will use in its operation and storing data.

4.2 Installation of October CMS

There are two ways of installing CMS:

4.2.1 Wizard installation

Wizard installation is straightforward to follow and do not need any specialized knowledge. (Octobercms.com, n.d.f)

1. First, prepare a server in the pc for the installation. In the MAMP environment, locate the htdocs folder inside MAMP application folder.
2. Download the installer achieve files and extract the installer file inside a folder for application inside htdocs folder of MAMP server.
3. Grant permission for installer and files if needed.
4. Locate and navigate the installation file and follow the instruction provided during the installation

4.2.2 Console Installation

Since October CMS is built on Laravel Framework, its console commands are constructed on Laravel's Artisan tool for development of the application. Laravel Artisan tool will be discussed briefly later. There are numerous CLI (command-line interface) commands and utilities that provide service to install, update, and robust development process. To perform installation process composer is needed to manage the dependencies of CMS and also can be achieved without a composer. First, we located the server location where the application to installed and then performed the installation.
We ran the following commands to begin the installation process with the help of composer we installed and created an October CMS project called IDNepal
Code Snippet 2. Command used to install October CMS project IDNepal.

```
composer create-project october/october IDNepal
```

After, downloading application source, we ran october:install command
Code Snippet 3. Command used to create database for the project.

```
php artisan october:install
```

And ran october:up for the migration in the database
Code Snippet 4. Command used for database migration.

```
php artisan october:up
```

And, inspectected config/database.php for more configuration if needed. (Octobercms.com. n.d.d)
After the installation process was over, there was the directory structure inside the project file as mentioned above in section directory structure October CMS.

4.3 Creating Default theme template

After Successful installation of October CMS platform, we started to build our websites. Using the user interface, we first created a new website template named "default". When template was created, the system automatically created the file structure that we described in the components of website development. Before starting the construction of the website, we acquired the jQuery and Bootstrap booklet online and included them together with our custom-made files in the assets folder.
In the default template using the twig filter "|theme" included all the styles and scripts that are subpages need for proper functions.

Code Snippet 5. Code to show how stylesheets are linked to the default theme template.

```
<link href="{{
    [
        'assets/lib/bootstrap/css/bootstrap.min.css',
        'assets/lib/animate/animate.min.css',
        'assets/lib/font-awesome/css/font-awesome.min.css',
        'assets/lib/ionicons/css/ionicons.min.css'
    ]|theme}}" rel="stylesheet">
```

Code Snippet 6. Code to show how JavaScript is linked to the theme template.

```
<script src="{{
    [
        'assets/lib/jquery/jquery.min.js',
        'assets/lib/bootstrap/js/bootstrap.bundle.min.js',
        'assets/lib/bootstrap/js/bootstrap.min.js',
        'assets/js/main.js'
    ]|theme }}" rel="text/javascript"></script>
```

In the default template, we included individual parts of the website that we develop independently. This allows the page and partial functions to include the entire section of the site or individual parts of the web page, such as navigation bar, header, footer, sections, main body, and so on as shown in Code Snippet 7.

Code Snippet 7. Code to show default page template for the website with partial page funtions included.

```
<header id="header">
        <div class="container">
            <div id="logo" class="pull-left">
                <h1><a href="#intro" class="scrollto">ID
Nepal</a></h1>
                <!-- <a href="#intro"><img src="img/logo.png" alt=""
title=""></a> -->
            </div>
            <!-- #nav-menu-container -->
            <nav id="nav-menu-container">
                {% partial "nav" %}
            </nav>
            <div class="clearfix"></div>
        </div>
        {% partial "bullet-nav" %}
    </header><!-- #header -->

    <!--==========================
    Intro Section
    ============================-->
    <section id="intro">
        {% partial "intro" %}
    </section><!-- #intro -->

    <main id="main">
        <!--===========================
    About Us Section
    ==============================-->
        <section id="about" class="section-bg">
            {% partial "about" %}
        </section><!-- #about -->
        <!--===========================
    Whywithus Section
    ==============================-->
        <section id="whywithus" class="section-bg">
            {% partial "whywithus" %}
        </section>

        <!--===========================
    our teams
    ==============================-->
        <section id="team" class="section-bg">
            {% partial "ourteam" %}
        </section>
        <!--===========================
    Contact Section
    ==============================-->
        <section id="timeline" class="section-bg">
            {% partial "timeline" %}
        </section>
        <!-- #contact form -->

        <section id="sendCV">
            <div class="container" style="display:flex;">
                {% component 'formComponent' %}
            </div>
        </section>

    </main>

    <!--===========================
    Footer
    ==============================-->
    <footer id="footer">
        {% partial "footer" %}
    </footer><!-- #footer -->
```

Due to specific implementation required by the content of the page, we created some plugin to meet the required functions.

4.4 Install/Creating a plugin

We installed/created Form, Timeline, Our team slider, and whywithus plugin using CLI interface that will publish customer hospitality features and controlled its views on the page by the user. The plugin can be installed in two different processes: one with the CLI interface. Here is an example.
Code Snippet 8. Code to install plugin called Timeline with Author as IDNepal.

```
php artisan plugin:install IDNepal.Timeline
or
php artisan create:plugin IDNepal.Timeline
```

4.5 Components of Plugins

We will talk about the component of the plugin briefly in this section. Components contain front-end views of the plugin and optional partials that might require for later uses in the plugin. Components are created using php artisan command create:component and followed by author name, plugin name and name of the component.(Octobercms.com. n.d.d)
Code Snippet 9. Command to create Timeline component for Timeline plugin.

```
php artisan create:component IDNeapl.Timeline TimelineCompo
```

4.5.1 Component Class Definition

The functionality and properties of components are defined in the component class file. The name of the component and the class of the component must be the same and extend CMS base component base class (i.e. \Cms\Classes\ComponentBase) (Octobercms.com. n.d.c).
For example,
Code Snippet 10. Example of functionality and properties defined in component class.

```
<?php namespace Idnepal\Timeline\Components;

use Cms\Classes\ComponentBase;
use Idnepal\Timeline\Models\Timeline;

class TimelineCompo extends ComponentBase
{
    public $timelines;
    public function componentDetails()
    {
        return [
            'name'        => 'Timeline Component',
            'description' => 'Timeline'
        ];
    }



    public function onRun(){
        $this->timelines = $this->_loadtimeline();
    }

    public function _loadtimeline(){
        if($this->property('noOfRow') > 0){
            return Timeline::take($this->property('noOfRow'))-
>orderBy('date','asc')->get();
        }
    }
}
```

componentDetails() method is required and returns name and description of the plugin that will be displayed in the CMS backend interface. The component will be available to the page or layout by attaching component variables, as short name or alias of the component which is defined in component registration method of the plugin.php file and components can be accessed using Twig notation as shown in the example.
Code Snippet 11. Example of accessing component in the page using twig notation.

```
description = "timeline"

[timeline]
==
{% component 'timeline' %}
```

4.5.2 Component Registration

We registered the component in the plugin.php file using the component register method to provide plugins functionality to the page or the layout. It provides the component's details and the alias name as a component variable.(OCM. n.d.a)
Code Snippet 12. Example of component registration in the plugin.php file.

```
<?php namespace Idnepal\Timeline;

use System\Classes\PluginBase;

class Plugin extends PluginBase
{
    public function registerComponents()
    {
        return[
            'Idnepal\Timeline\Components\TimelineCompo' =>
'timeline'
        ];
    }

    public function registerSettings()
    {
    }
}
```

4.5.3 Component Properties

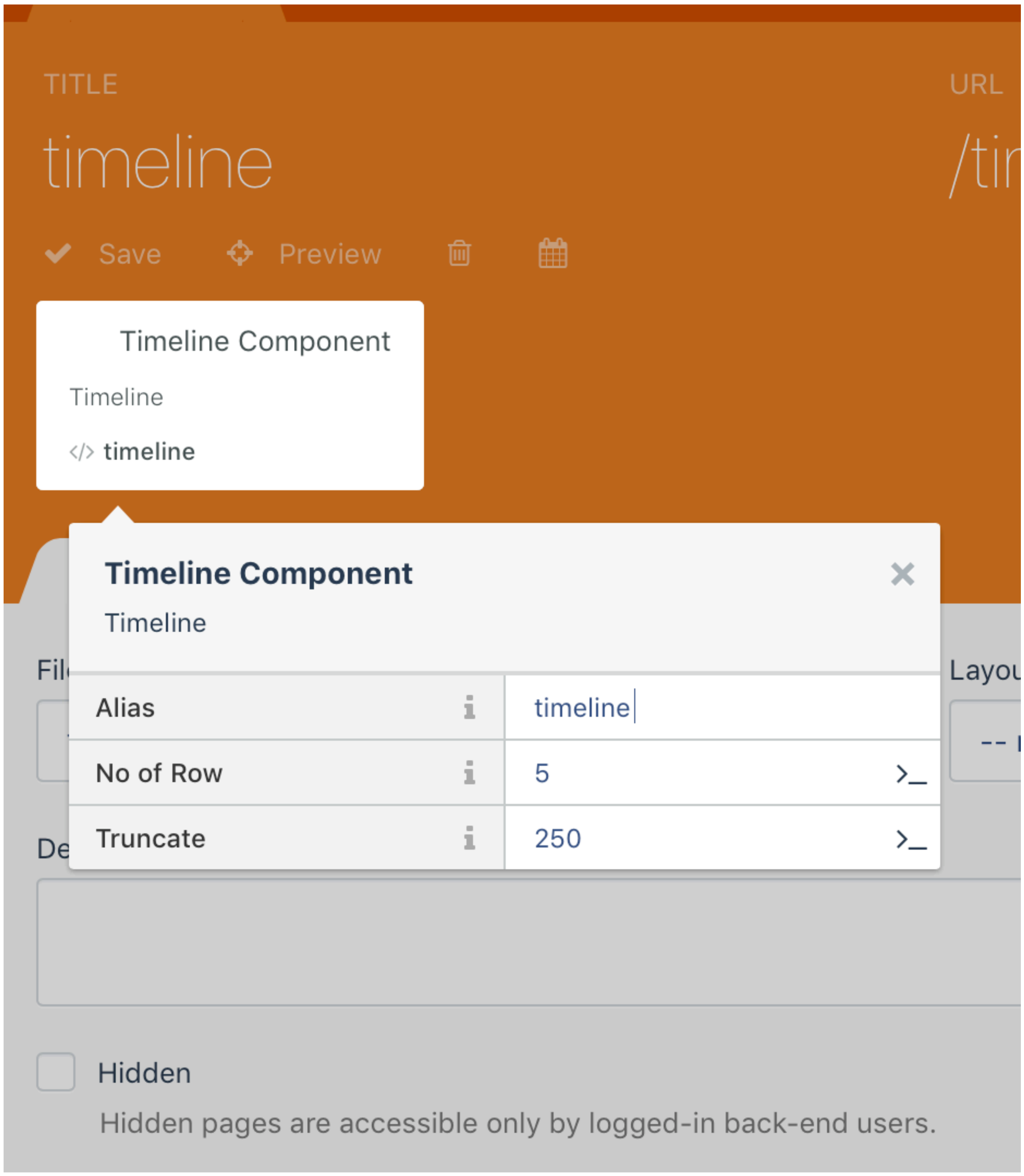
The properties of the components are defined in the component class which provides the configuration functionality and control over component page layout.(Octobercms.com. n.d.c)
Code Snippet 13. Example of defining properties in component class.

```
public function defineProperties()
    {
        return [
            'noOfRow' => [
                'title' => 'No of Row',
                'description' => 'no of row to display',
                'default' => '5',
                'validationPattern' => '^[0-9]+$',
                'validationMessage' => 'Only numbers are allowed'
            ],
            'truncate' => [
                'title' => 'Truncate',
                'description' => 'display no of characters',
                'default' => '250',
            ]

        ];
    }

            //readmore text options
            'readmoreTextOption' => [
                'title' => 'readmore options',
                'description' => 'readmore text option at the end of
excerpt',
                'default' => 'readmore',
                'type' => 'string'
            ]
        ];
    }
```

The above example shows how to we configure and add more control to the content of the timeline to display on the page. The configuration interface is also available in the backend of the page as shown in Figure 10.



Timeline Component

Timeline

Alias	timeline
No of Row	5
Truncate	250

Figure 10. Backend control for the post in the page.

4.6 Model of Timeline Plugin

After the component, we created a model called Timeline for the plugin Timeline using the "Builder" plugin to create a table called idnepal_timeline_timeline to store the content of the timeline in the database as shown in figure 11. We also determined its types, length and properties required for individual columns.

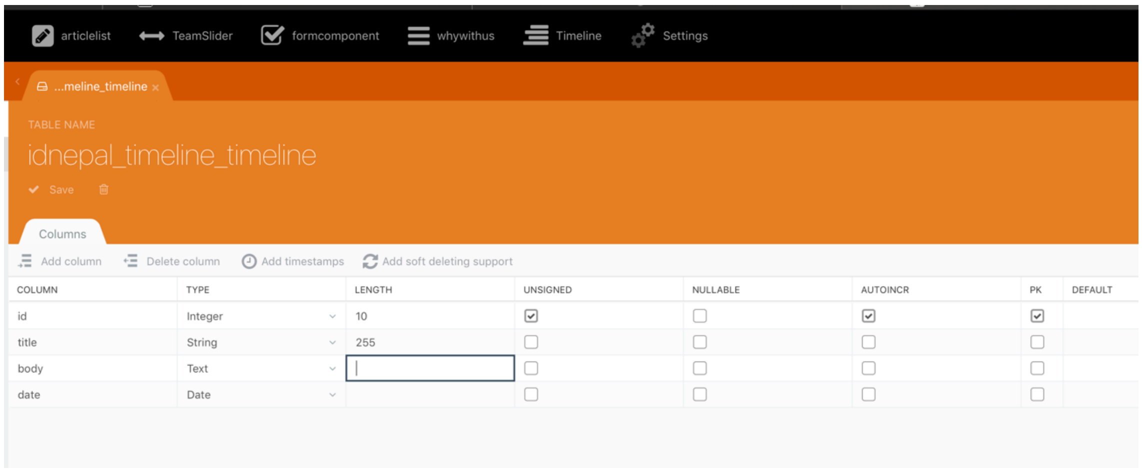


TABLE NAME
idnepal_timeline_timeline

COLUMN	TYPE	LENGTH	UNSIGNED	NULLABLE	AUTOINCR	PK	DEFAULT
id	Integer	10	✓	☐	✓	✓	
title	String	255	☐	☐	☐	☐	
body	Text		☐	☐	☐	☐	
date	Date		☐	☐	☐	☐	

Figure 11.Backend view of the database table in builder plugin.

In the next step, we created two sub-forms called Forms and List appeared inside the model. The first forms tab we defined the fields.yaml fields, for required controls, to insert data in the table: we used “text” field for the title of the agendas, "richeditor" for the description of the body, and "date picker" for the date as shown in Figure 12.

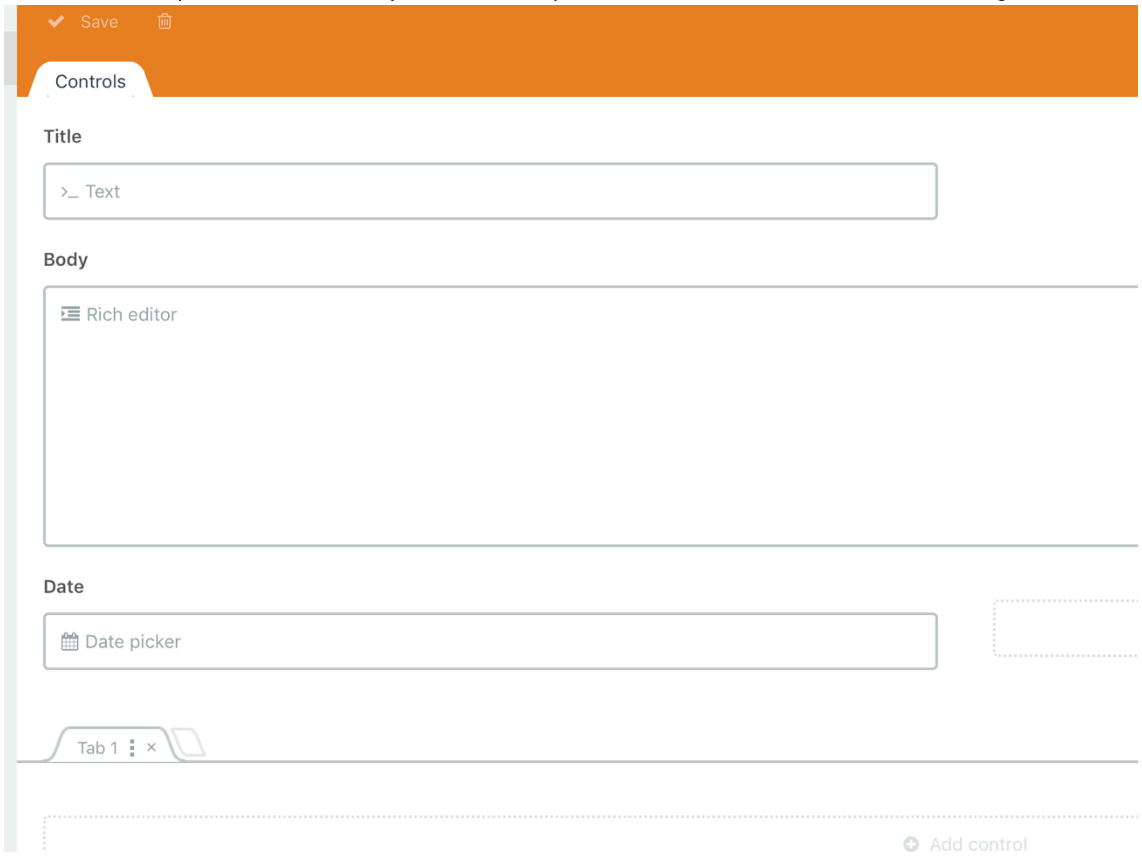
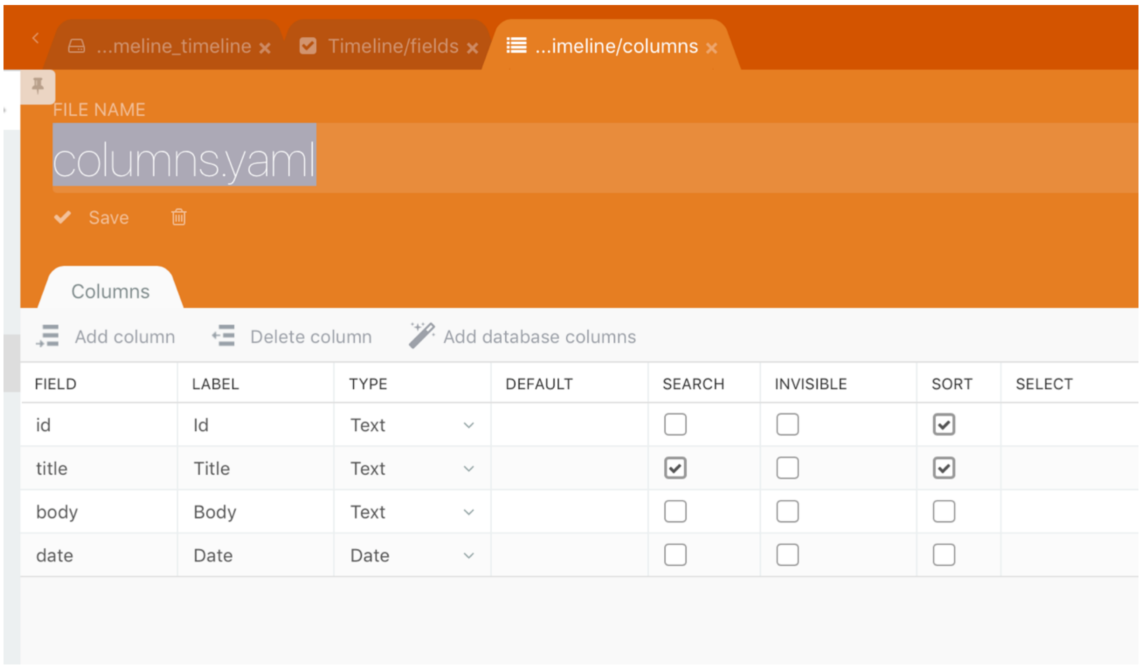


Figure 12. Input Controller for the timeline plugin in the backend.

In the second tab Lists, we created a “columns.yaml” list through which we can access all content entries made using input fields as in Figure 13 and linked them with the table created in the database.



FILE NAME
columns.yaml

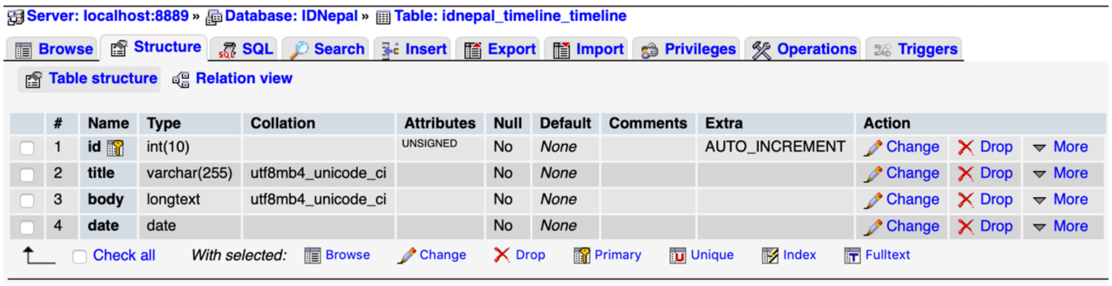
FIELD	LABEL	TYPE	DEFAULT	SEARCH	INVISIBLE	SORT	SELECT
id	Id	Text		☐	☐	✓	
title	Title	Text		✓	☐	✓	
body	Body	Text		☐	☐	☐	
date	Date	Date		☐	☐	☐	

Figure 13. Required fields to form a timeline component at the backend user interface.

After saving the table OctoberCMS, it automatically generated necessary migration to the database.
Code Snippet 14. Automatically generated code for migration to the database.

```
<?php namespace Idnepal\Timeline\Updates;
use Schema;
use October\Rain\Database\Updates\Migration;
class BuilderTableCreateIdnepalTimelineTimeline extends Migration
{
    public function up()
    {
        Schema::create('idnepal_timeline_timeline', function($table)
        {
            $table->engine = 'InnoDB';
            $table->increments('id')->unsigned();
            $table->string('title');
            $table->string('body');
            $table->date('date');
        });
    }
    public function down()
    {
        Schema::dropIfExists('idnepal_timeline_timeline');
    }
}
```

Figure 14 shows database created by the Code Snippet 13 in the server.



Server: localhost:8889 » Database: IDNepal » Table: idnepal_timeline_timeline

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Table structure | Relation view

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	**id**	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT	Change	Drop	More
☐	2	**title**	varchar(255)	utf8mb4_unicode_ci		No	*None*			Change	Drop	More
☐	3	**body**	longtext	utf8mb4_unicode_ci		No	*None*			Change	Drop	More
☐	4	**date**	date			No	*None*			Change	Drop	More

☐ Check all *With selected:* Browse Change Drop Primary Unique Index Fulltext

Figure 14. Table created in the database, view from phpMyAdmin.

In the backend, we created a new menu tab to access all records of model data of the database. We have set a new icon for the menu to identify the tab of the plugin just created as shown in Figure 15.

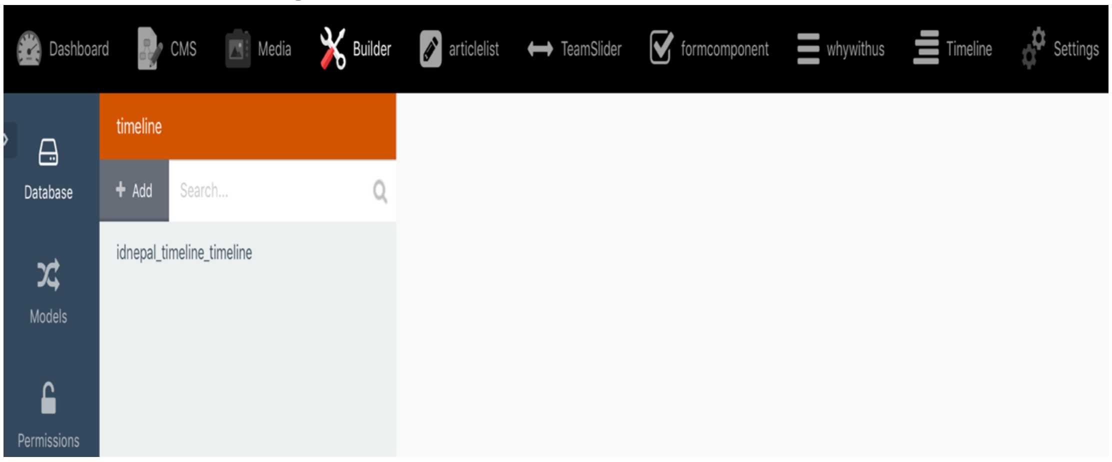
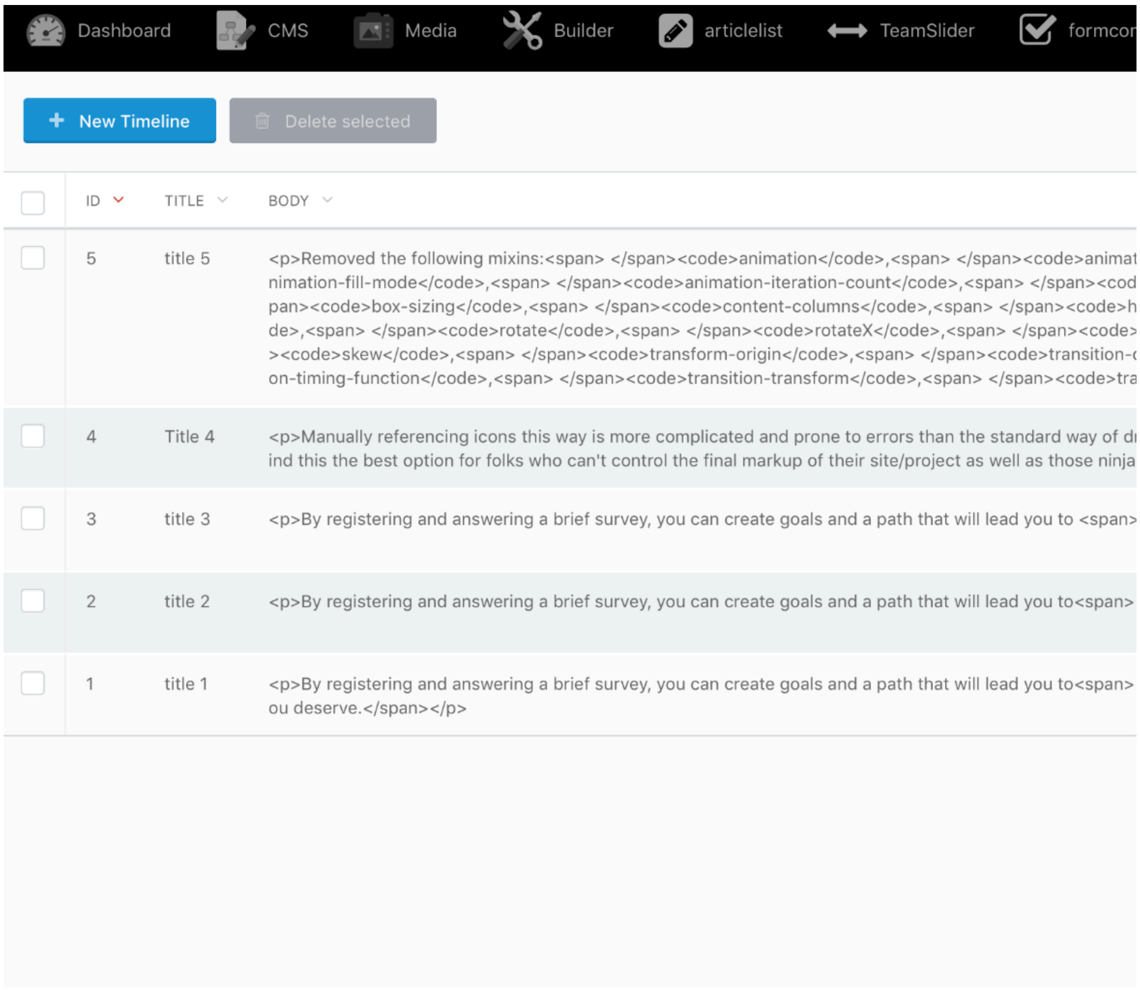


Figure 15. List of plugins created for the website on the top navigation of the backend page.

The menu “timeline” we created enable us to access timeline model and all the controls, forms which created to enter new data, list of the data entered, delete data and edit data. Figure 16 shows the number of entries made from the input controls from Figure 13.



	ID	TITLE	BODY
	5	title 5	<p>Removed the following mixins:<span> </span><code>animation</code>,<span> </span><code>animat nimation-fill-mode</code>,<span> </span><code>animation-iteration-count</code>,<span> </span><cod pan><code>box-sizing</code>,<span> </span><code>content-columns</code>,<span> </span><code>h de>,<span> </span><code>rotate</code>,<span> </span><code>rotateX</code>,<span> </span><code> ><code>skew</code>,<span> </span><code>transform-origin</code>,<span> </span><code>transition-(on-timing-function</code>,<span> </span><code>transition-transform</code>,<span> </span><code>tra
	4	Title 4	<p>Manually referencing icons this way is more complicated and prone to errors than the standard way of di ind this the best option for folks who can't control the final markup of their site/project as well as those ninja
	3	title 3	<p>By registering and answering a brief survey, you can create goals and a path that will lead you to <span>
	2	title 2	<p>By registering and answering a brief survey, you can create goals and a path that will lead you to<span>
	1	title 1	<p>By registering and answering a brief survey, you can create goals and a path that will lead you to<span> ou deserve.</span></p>

Figure 16. Timeline backend view after inserting data for timeline component.

4.7 Controller of the plugin

In the Controller tab, we created Timeline controller for the plugin to define the base model and active timeline menu. By creating the controller, we connected our base model and the menu controller for the Timeline plugin to provide more functionality and control over necessary components related to the plugin.
While creating a controller, we enabled the list and form behaviour of the plugin which provides information about the model connected and URL of the configuring files of the controller as shown in Figure 17.

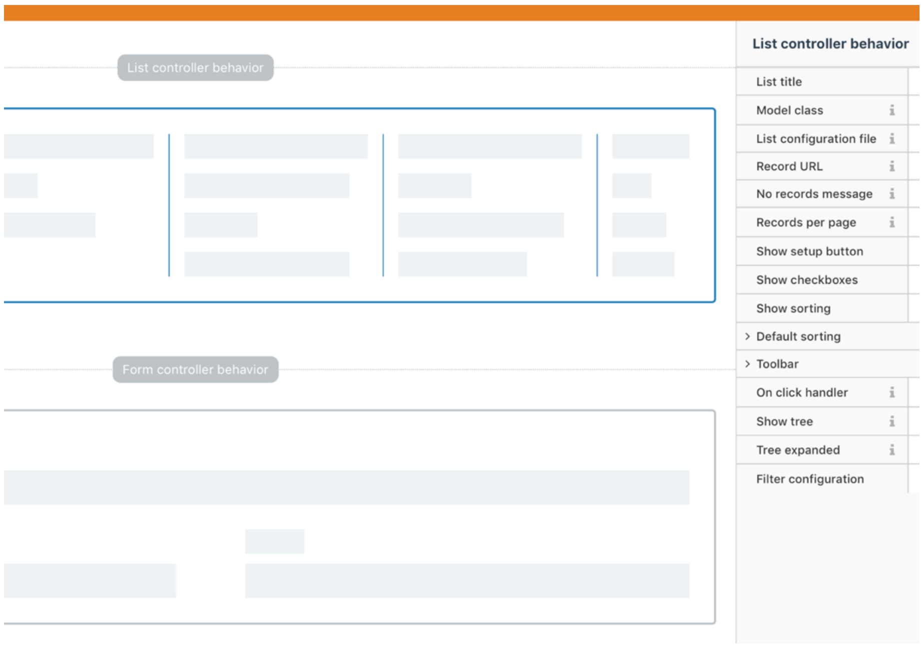
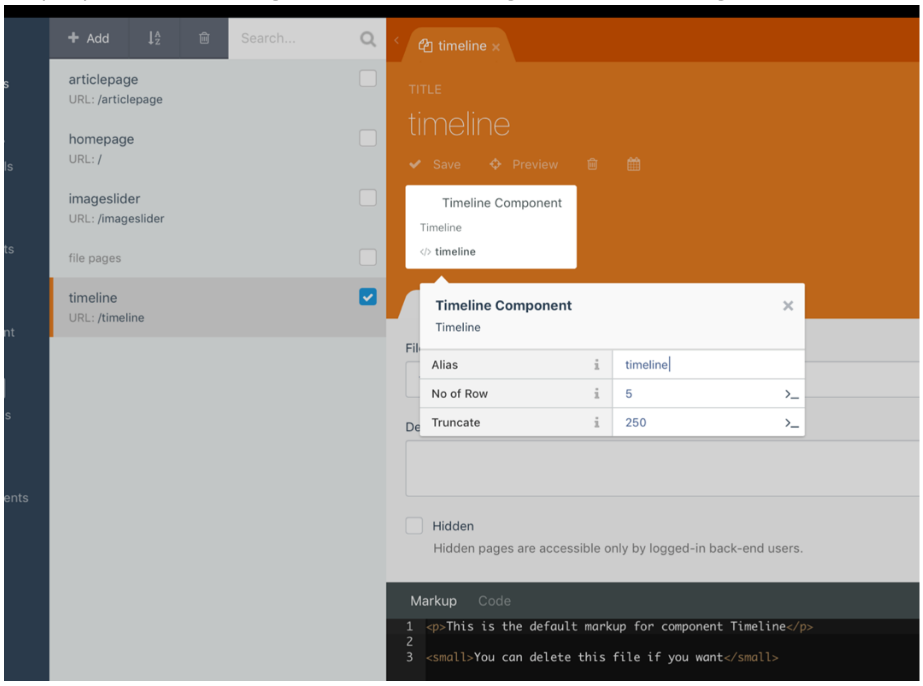


Figure 17. The behaviour of the controller in the backend.

4.8 Use of plugin

We used the plugin in the default layout of the website to show the dynamic road map of the company. We used the plugin in the layout by using Twig command "{% component 'timeline' %}" in the timeline partial page and then called this partial page to the default layouts using {% partial ‘timeline’ %} where the timeline will be displayed as in the figure 19 with configuration form Figure 18.

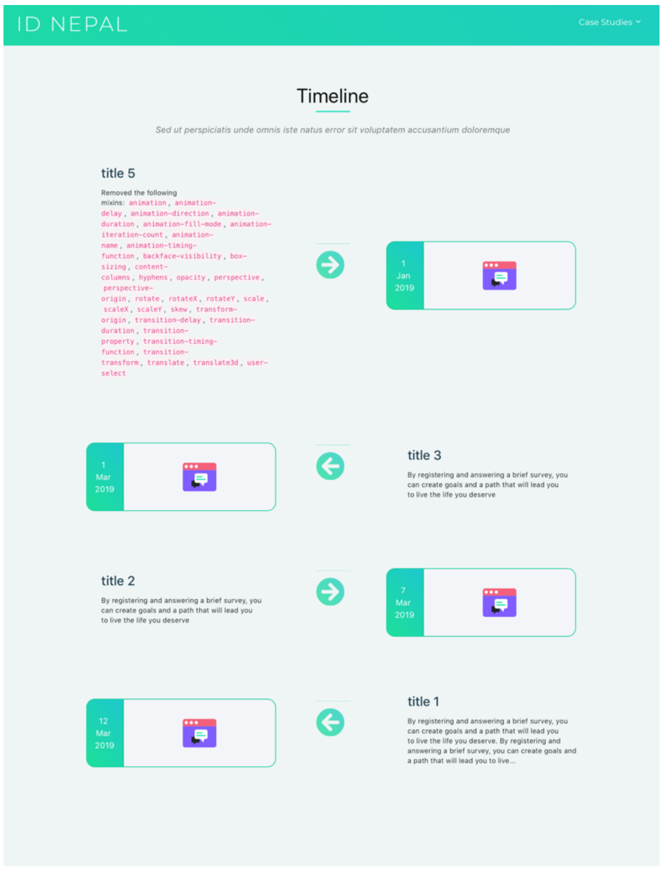


Alias	timeline
No of Row	5
Truncate	250

```
<p>This is the default markup for component Timeline</p>

<small>You can delete this file if you want</small>
```

Figure 18. Shows the config inspector of the timeline component.



Timeline

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque

title 5

Removed the following mixins: `animation`, `animation-delay`, `animation-direction`, `animation-duration`, `animation-fill-mode`, `animation-iteration-count`, `animation-name`, `animation-timing-function`, `backface-visibility`, `box-sizing`, `content-columns`, `hyphens`, `opacity`, `perspective`, `perspective-origin`, `rotate`, `rotateX`, `rotateY`, `scale`, `scaleX`, `scaleY`, `skew`, `transform-origin`, `transition-delay`, `transition-duration`, `transition-property`, `transition-timing-function`, `transition-transform`, `translate`, `translate3d`, `user-select`

1 Jan 2019

1 Mar 2019

title 3

By registering and answering a brief survey, you can create goals and a path that will lead you to live the life you deserve

title 2

By registering and answering a brief survey, you can create goals and a path that will lead you to live the life you deserve

7 Mar 2019

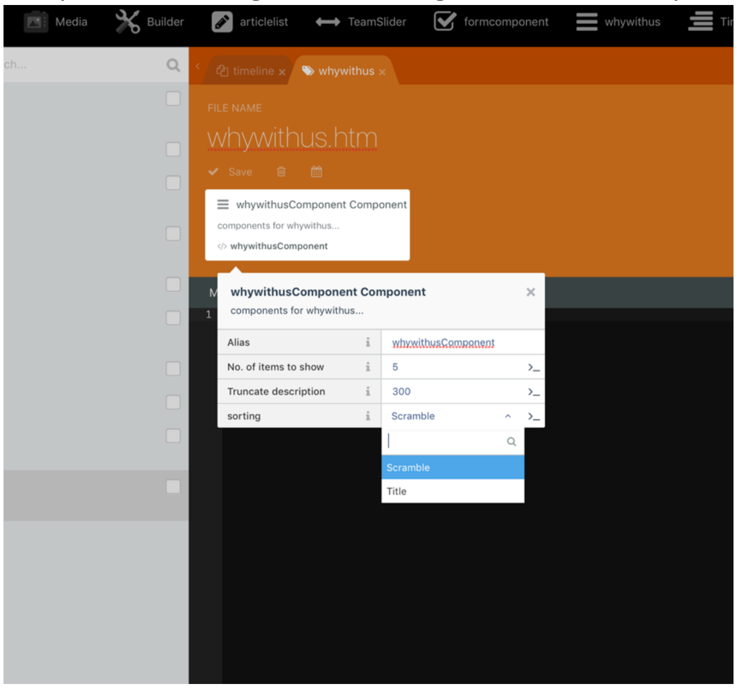
12 Mar 2019

title 1

By registering and answering a brief survey, you can create goals and a path that will lead you to live the life you deserve. By registering and answering a brief survey, you can create goals and a path that will lead you to live...

Figure 19. After timeline partial called in layout.

We tested more with different plugins on the page with different properties of the component as in Figure 20 and Figure 21 shows output of the configuration.



whywithus.htm

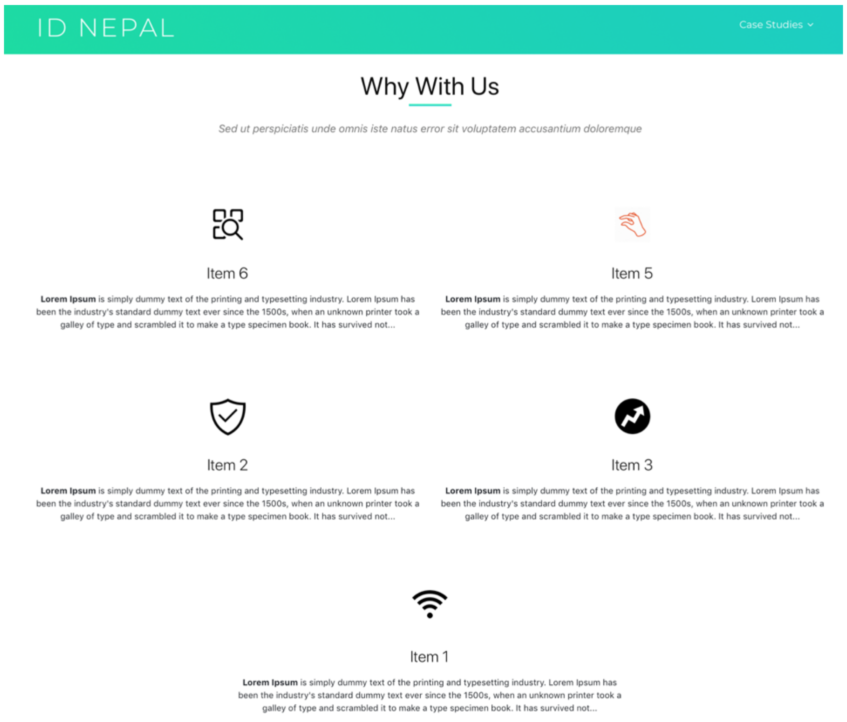
whywithusComponent Component
components for whywithus...
whywithusComponent

whywithusComponent Component
components for whywithus...

Alias	whywithusComponent
No. of items to show	5
Truncate description	300
sorting	Scramble

Scramble
Title

Figure 20. Different option from component properties.



ID NEPAL

Case Studies

Why With Us

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque

Item 6

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not...

Item 5

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not...

Item 2

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not...

Item 3

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not...

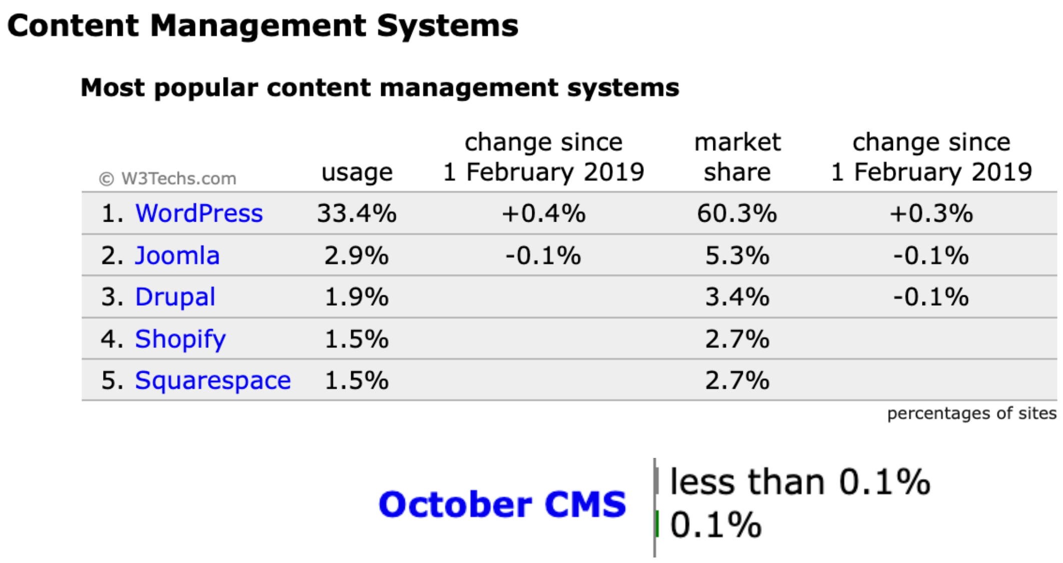
Item 1

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not...

Figure 21. Using sorting config and the number of items displayed is 5 configured in the property inspector.

5 COMPARISONS

Presently, the CMS is dominated and controlled by CMS such as WordPress, Joomla!, and Drupal.
According to the survey of w3techs in Figure 22, WordPress dominated the market share of CMS by60.3% and followed by Joomla! And Drupal. And October CMS is far below somewhere in the table having the market share of less than 1% used in websites as CMS until 1 February 2019. (W3techs.com, 2019)


Content Management Systems

Most popular content management systems

© W3Techs.com	usage	change since 1 February 2019	market share	change since 1 February 2019
1. WordPress	33.4%	+0.4%	60.3%	+0.3%
2. Joomla	2.9%	-0.1%	5.3%	-0.1%
3. Drupal	1.9%		3.4%	-0.1%
4. Shopify	1.5%		2.7%	
5. Squarespace	1.5%		2.7%	

percentages of sites

October CMS	less than 0.1%
	0.1%

Figure 22. Survey of CMS by w3techs.

Despite the market share, developer around the world adopting OctoberCMS as CMS for their project. Since, October CMS first release received 8,279 stars (GitHub, n.d.c), WordPress got 12,133 stars (GitHub, n.d.d), Joomla! got 3,210 stars (GitHub, n.d.b), and Drupal got 3,213 stars (GitHub, n.d.a) till the date 11 March 2019 which shows the increasing popularity among the developers.

Here are some advantages and disadvantages of popular CMS on the market:
Table 2. Advantages and Disadvantages of Wordpress (Erickson, 2017).

WordPress	
Advantages	Disadvantages
Free and open source	Blog centric
Lots of free themes	Plugin dependent
Free hosting under WordPress subdomain	Untested plugins, the threat of vulnerability
User-friendly	More plugins more load on performance
Plugins help for SEO	For the database, plugin required
Post level privacy control	Insecure and outdated plugins are available

Table 3. Advantages and Disadvantages Of Joomla! (All Things Internet | Informatics Inc., 2014b).

Joomla!	
Advantages	Disadvantages
Free and opensource	Outdated and untested plugins available
Lots of plugin and extensions available	Page rendering may take long
User-friendly interface	Limited market place
Plugin management for the unnecessary plugin	
Highly customizable	
Flexible in editing contents	

Table 4. Advantages and Disadvantages. Of Drupal (All Things Internet | Informatics Inc., 2014a).

Drupal	
Advantages	Disadvantages
Free and open source	Lack of free modules and themes
Good modular directory	Hard to use, require right technical expertise
Developer friendly	
Capable of producing advanced sites with complex functionality	
Good for community platform sites	

5.1 Advantages and Disadvantages

October CMS is created to overcome the weakness of other CMS which have dominated the market like WordPress, Joomla!, and Drupal. Like WordPress, Joomla!, and Drupal is also open source goes hand to hand with market dominated CMS.

October CMS is a newbie in the CMS market but has a lot of features that attracted developers gained popularity in less duration of time around the world in comparison of others.
Below are some features that make October CMS more potent than other CMS available.

- Simplicity – October CMS is built to reduce developer pain by doing most of the dirty jobs like handling authentication functionality, creating pages, components, partials by couple lines of code and maintain directory in systematic and straightforward way so that we can debug in no time, and easily components are duplicated by copying and pasting wherever we like in the web page or application.
- Clean syntax – October CMS is built on the top of Laravel and uses Twig Templating syntax that makes code stay clean.
- Flat file system – October CMS was voted best flat file CMS in CMS critic award contest in 2018(CMS, 2018). It uses a flat file system to serve website structure providing the ability to cache and optimize the application for better performance.
- In-Built Authentication and Authorization - October CMS has in-built authentication and Authorization functions from its first beta release which reduced ache of developers by handling complex logical functions required by its own.
- Regular Updates - The community is growing day by day which provides more expertise supports and releases regular updates as needed.
- Secured - Laravel is the core framework of October CMS and has a unique filing system that provides only access of index.php page to avoid forgery and plugins are approved and ensured before listing to the market. (Leader Internet, 2017a)
- Performance - Laravel has by default minification function to minify files and store in plain text for better caching.(Leader Internet, 2017a)

Despite great features, there are some drawbacks of using October CMS

- Not end user-friendly
  To use October CMS, one should have at least basic knowledge of Web Development.
- Developer focused (Tobies, 2019)
- Less plugin than other CMS (Leader Internet, 2017b)

6 CONCLUSION

In this thesis, we used the practical example of web development to demonstrate the use and operation of the October CMS platform. We first introduced and described the features of the most common content management systems to be familiar with the function and purpose of the content management system. We have found that the use of CMS systems is not limited to a specific group of users since different organizations and companies use this CMS in their operation. And, we also found that the advantages and disadvantages within the same areas depend primarily on individual content management systems.

We described the Laravel framework, on the basis of which was built to understand the operation and ecosystem of October CMS platform. In brief, we also presented the MVC architecture which allows the modular development of web pages and its plugins. We developed a plugin for a website to display and manage the content of the page. The October CMS user-friendly backend and CLI interface helps to create the plugin at a rapid speed and modified as our need. We are not able to test each feature of October CMS during production. Nevertheless, we can confirm during the development process of the plugin, and we realized the basis of this platform, which speeds up the development of the web application.

We delivered the product after developing the website on the October CMS which gives the overview of the company, agendas, future initiatives and innovation by developing the different plugin for the sites like form, team slider, timeline and so on.

REFERENCES

All Things Internet | Informatics Inc. (2014a). *Pros and Cons of Drupal CMS | Informatics Inc.*. [online] Available at: https://www.informaticsinc.com/blog/2014/pros-and-cons-drupal-content-management-system [Accessed 10 Mar. 2019].

All Things Internet | Informatics Inc. (2014b). *Pros and Cons of Joomla! | Informatics Inc.*. [online] Available at: https://www.informaticsinc.com/blog/2014/pros-and-cons-joomla-content-management-system [Accessed 10 Mar. 2019].

archybold.com. (2015). *Laravel to October CMS: The Differences (Part 1) | The Blog*. [online] Available at: https://www.archybold.com/blog/post/laravel-vs-october-cms-differences-part-1 [Accessed 6 Mar. 2019].

Barker, D. (2016). *Web Content Management*. [online] O'Reilly | Safari. Available at: https://www.oreilly.com/library/view/web-content-management/9781491908112/ch01.html [Accessed 28 Mar. 2019].

Laminack, B.I. (2001). *A Brief History of Content Management Systems*. [online] Laminack.com. Available at: http://www.laminack.com/index.php/technology/11-a-brief-history-of-content-management-systems [Accessed 18 Mar. 2019].

CMS, O. (2015). *Putting OctoberCMS into words - October CMS*. [online] Octobercms.com. Available at: https://octobercms.com/blog/post/putting-octobercms-words [Accessed 6 Mar. 2019].

CMS, O. (2018). *October CMS voted the Best Flat File CMS in 2018 - October CMS*. [online] Octobercms.com. Available at: https://octobercms.com/blog/post/october-cms-voted-best-flat-file-cms-2018 [Accessed 10 Mar. 2019].

CMS, O. (n.d.a). *Registration - October CMS*. [online] Octobercms.com. Available at: https://octobercms.com/docs/plugin/registration [Accessed 6 Mar. 2019].

CMS, O. (n.d.b). *Templating - October CMS*. [online] Octobercms.com. Available at: https://octobercms.com/docs/markup/templating [Accessed 13 Mar. 2019].

CMS, O. (n.d.c). *Version history - October CMS*. [online] Octobercms.com. Available at: https://octobercms.com/docs/plugin/updates [Accessed 6 Mar. 2019].

Cms.co.uk. (n.d.). *For Newbies*. [online] Available at: http://www.cms.co.uk/for-newbies.html [Accessed 15 Mar. 2019].

CodinGame. (n.d.). *Coding Games and Programming Challenges to Code Better*. [online] Available at: https://www.codingame.com/playgrounds/5886/introduction-to-php-website-development/what-is-jquery [Accessed 13 Mar. 2019].

En.wikipedia.org. (n.d.). *Plug-in (computing)*. [online] Available at: https://en.wikipedia.org/wiki/Plug-in_(computing) [Accessed 6 Mar. 2019].

Erickson, C. (2017). *The pros and cons of choosing WordPress for your website CMS*. [online] Arcstone.com. Available at: https://www.arcstone.com/blog/pros-and-cons-of-wordpress-cms [Accessed 10 Mar. 2019].

GitHub. (n.d.a). *drupal/drupal*. [online] Available at: https://github.com/drupal/drupal [Accessed 10 Mar. 2019].

GitHub. (n.d.b). *joomla/joomla-cms*. [online] Available at: https://github.com/joomla/joomla-cms [Accessed 10 Mar. 2019].

GitHub. (n.d.c). *octobercms/october*. [online] Available at: https://github.com/octobercms/october [Accessed 10 Mar. 2019].

GitHub. (n.d.d). *WordPress/WordPress*. [online] Available at: https://github.com/WordPress/WordPress [Accessed 10 Mar. 2019].

Ighodaro, N. (2018). *How Laravel implements MVC and how to use it effectively*. [online] Pusher Blog. Available at: https://blog.pusher.com/laravel-mvc-use/ [Accessed 5 Mar. 2019].

Leader Internet. (2017a). *October CMS vs Drupal*. [online] Available at: https://leaderinternet.com/blog/october-cms-vs-drupal [Accessed 10 Mar. 2019].

Leader Internet. (2017b). *October CMS vs Wordpress*. [online] Available at: https://leaderinternet.com/blog/october-cms-vs-wordpress [Accessed 10 Mar. 2019].

Nilanchala (2017). *A Brief Introduction to Laravel PHP Framework Features and Version History | Stacktips*. [online] Stacktips - How-to Guides, Tutorials and Code Snippets. Available at: https://stacktips.com/laravel/intro-to-laravel-php-framework-and-features [Accessed 4 Mar. 2019].

Octobercms.com. (n.d.a). *About - October CMS*. [online] Available at: https://octobercms.com/about [Accessed 28 Mar. 2019].

Octobercms.com. (n.d.b). *Builder plugin - October CMS*. [online] Available at: https://octobercms.com/plugin/rainlab-builder [Accessed 6 Mar. 2019].

Octobercms.com. (n.d.c). *Building components - October CMS*. [online] Available at: https://octobercms.com/docs/plugin/components [Accessed 7 Mar. 2019].

Octobercms.com. (n.d.d). *Command list - October CMS*. [online] Available at: https://octobercms.com/docs/console/commands#console-install [Accessed 7 Mar. 2019].

Octobercms.com. (n.d.e). *Controllers & AJAX - October CMS*. [online] Available at: https://octobercms.com/docs/backend/controllers-ajax [Accessed 6 Mar. 2019].

Octobercms.com. (n.d.f). *Installation - October CMS*. [online] Available at: https://octobercms.com/docs/setup/installation [Accessed 7 Mar. 2019].

Octobercms.com. (n.d.g). *Models - October CMS*. [online] Available at: https://octobercms.com/docs/database/model [Accessed 6 Mar. 2019].

Otwell, T. (n.d.). *Directory Structure - Laravel - The PHP Framework For Web Artisans*. [online] Laravel.com. Available at: https://laravel.com/docs/master/structure [Accessed 5 Mar. 2019].

Rizwan, S. (2018). *Clash Between Top Laravel CMS: OctoberCMS VS AsgardCMS*. [online] The Official Cloudways Blog. Available at: https://www.cloudways.com/blog/asgardcms-vs-octobercms/ [Accessed 4 Mar. 2019].

Slant, 1. (n.d.). *Slant - 13 Best PHP CMS as of 2019*. [online] Slant. Available at: https://www.slant.co/topics/5409/~php-cms [Accessed 10 Mar. 2019].

Smooke, D. (2017). *Evolution of the Content Management System and the CMS API*. [online] Hacker Noon. Available at: https://hackernoon.com/evolution-of-the-content-management-system-and-the-cms-api-bfdfff86fc5b [Accessed 18 Mar. 2019].

Surguy, M. (2013). *History of Laravel PHP framework, Eloquence emerging - Maks Surguy's blog on Technology Innovation, IoT, Design and Code*. [online] Maks Surguy's blog on Technology Innovation, IoT, Design and Code. Available at: https://maxoffsky.com/code-blog/history-of-laravel-php-framework-eloquence-emerging/ [Accessed 5 Mar. 2019].

Tobies, V. (2019). *October CMS Alternatives | Reviews | Pros & Cons - Alternative.me*. [online] Alternative.me. Available at: https://alternative.me/october-cms [Accessed 5 Mar. 2019].

Tutorialsteacher.com. (n.d.). *MVC Architecture*. [online] Available at: https://www.tutorialsteacher.com/mvc/mvc-architecture [Accessed 5 Mar. 2019].

W3schools.com. (n.d.a). *Bootstrap 4 Get Started*. [online] Available at: https://www.w3schools.com/bootstrap4/bootstrap_get_started.asp [Accessed 28 Mar. 2019].

W3schools.com. (n.d.b). *CSS Introduction*. [online] Available at: https://www.w3schools.com/css/css_intro.asp [Accessed 13 Mar. 2019].

W3schools.com. (n.d.c). *Introduction to HTML*. [online] Available at: https://www.w3schools.com/html/html_intro.asp [Accessed 13 Mar. 2019].

W3techs.com. (2019). *Usage Statistics and Market Share of Content Management Systems, March 2019*. [online] Available at: https://w3techs.com/technologies/overview/content_management/all [Accessed 10 Mar. 2019].

www.tutorialspoint.com. (n.d.). *MVC Framework Introduction*. [online] Available at: https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm [Accessed 5 Mar. 2019].