

Bachelor thesis (TUAS)

Degree Program in Information Technology

NINFOS11

2016

Shovit Thapa

MODERN WEB ELEMENTS, DEVELOPMENT METHODS AND TEHCNIQUES



TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree programme | Information Technology

April 2016 | 43

Instructor: Al-Bermanei Hazem

Shovit Thapa

MODERN WEB ELEMENTS, DEVELOPMENT METHODS AND TECHNIQUES

Most modern websites have complex design, appealing contents, include many features and services, and blend with any screens and browsers. Developing such websites requires deeper understanding of modern web elements and implementation of complex techniques. With the advancement of technology, methods and techniques of modern web development are advancing too. Hence, a web developer has to be competent enough to understand and use modern techniques and deploy all the required modern web elements in the websites.

The main objective of the thesis is to explore the major elements of modern web development and how they can be implemented resourcefully. The thesis also emphasizes the importance of opting for modern techniques for faster, better and robust web development.

In this thesis, few of the modern web development methods applied in designing three websites <http://geosmartcity.dc.turkuamk.fi/>, <http://geosmartcity.dc.turkuamk.fi/leaflet/index.html> and lavaplat.dc.turkuamk.fi/public are discussed in detail. The thesis also discusses a detailed research into various components and elements of modern websites with working codes where applicable.

KEYWORDS:

User Interface, Responsiveness, Multilingualism, User Experience, Laravel, CRUD functions.

CONTENTS

LIST OF PICTURES

LIST OF TABLES

LIST OF ABBREVIATIONS (OR) SYMBOLS

1 INTRODUCTION	7
2 BACK-END AND FRONT-END DEVELOPMENT	9
3 MODERN WEB ELEMENTS	11
3.1 User Experience	12
3.1.1 Colour	13
3.1.2 Graphics	13
3.1.3 Usability	12
3.2 Carousel	14
3.3 Parallax Scrolling	15
3.4 Scroll Reveal	16
3.5 Transition, transform and animation	16
3.5.1 Transition	16
3.5.2 Transform	17
3.5.3 Animation	18
3.6 Responsiveness	20
3.6.1 Fluid layout	20
3.6.2 Resizable image	21
3.6.3 CSS Media Query	21
3.6.4 Responsiveness in Bootstrap	24
3.7 Multilingualism	24
4 RESOURCEFUL WEB DEVELOPMENT	29
4.1 Static websites	31
4.1.1 Out-of-the box features	30
4.1.2 Maintenance	30
4.1.3 Extensibility	30
4.2 Dynamic websites	31
4.2.1 Laravel – a PHP framework	32
5 CONCLUSION	40
REFERENCES	42

LIST OF PICTURES

Figure 1. Front-End vs Back-End	10
Figure 2. Kuler colour Palette	12
Figure 3. A full page carousel	14
Figure 4. A chart for CSS3 animation browser supports	20
Figure 5. Mobile portrait view and tablet landscape view	21
Figure 6. File structure of GeosmartCity website	25
Figure 7. Content of language directory	26
Figure 8. WordPress dashboard for adding widgets	30
Figure 9. Core relation between Model, Controller and View	33
Figure 10. Registration form displayed by view	35
Figure 11. Output of the URL example.com/user	38

LIST OF TABLES

Table 1. Achieving a simple carousel in JavaScript	15
Table 2. Transition property in CSS	17
Table 3. Transform property in CSS	18
Table 4. Defining keyframes in CSS	18
Table 5. Defining transform property in keyframes	19
Table 6. Calling animation property inside the div element	19
Table 7. Defining the container's width in percentage	21
Table 8. Meta tag in HTML	22
Table 9. Media query defined in CSS	22
Table 10. Media rule in CSS	23
Table 11. init.php file	26
Table 12. Loading a page based on session	27
Table 13. GET and SESSION variables from URL	28

Table 14. Defining the resource controller in Laravel	33
Table 15. List of routes generated by RESTful routing	34
Table 16. Display the register View from Controller	35
Table 17. User data is validated and stored from the Controller	36
Table 18. User Model class	37
Table 19. User's data is passed from the Controller to the View	37
Table 20. Users View page	38

LIST OF ABBREVIATIONS (OR) SYMBOLS

HTML	Hyper Text Markup Language
CSS	Cascading Style Sheet
PHP	Pre Processor Hypertext
URL	Uniform Resource Locator
UI	User Interface
UX	User Experience
px	pixel
2D	2 Dimensional
3D	3 Dimensional
CMS	Content Management System
MVC	Model-View-Controller
WF	Web Framework
WAF	Web Application Framework
CRUD	Create, Read, Update, and Delete

1 INTRODUCTION

With the advent of internet and digital media, websites have become an indispensable part of our information system. To reach a large number of potential customers, companies expose their company profiles and advertise services via websites. Any organisation, educational or even governmental, releases data, vital information and instructions to thousands of people through websites. Similarly, e-news portals, online portfolios, online marketing are few examples to augment how heavily we rely on websites these days. In fact, websites have become the key source of information on the internet and its popularity is ever growing.

As the popularity of website is growing, the techniques and methods behind the web development are also rapidly advancing. The sites end-users browse these days are not as simple and limited as they used to be few years back. Modern web development tools are different, designs and looks of the sites are fairly superior and the requirements and services to be included are complex to develop. Designing websites have become even more challenging because of variant devices and the wide range of screen sizes and different browsers. For a web developer, it is vital to understand and adapt the modern trend and techniques in developing professional websites to meet clients' and end-users' requirements.

Similarly, variant web elements have been introduced in modern web development to enhance the look and feel of websites. Colour and graphics are profoundly used to express and emphasize the meaning of web contents. Images are artistically presented in various forms, like carousel and parallel scrolling. In addition, animations, transition and transform have become common and important parts of modern web elements. While these kinds of elements are a key to designing modern websites, they add complexity to the development process. So a developer should be well aware of all such requirements and should be able to implement them whenever required.

Additionally, one of the most important aspects of developing a modern website is resourceful development. Modern organisational or business websites have

huge requirements. Developing such websites in a traditional way needs a huge amount of time and resources. So, to overcome these issues, developers can opt for the modern web development tools available. These tools are intended to make the web development process faster, effective and more profitable.

2 FRONT-END AND BACK-END DEVELOPMENT

Web development comprises front-end and back-end development [1]. Front-end, often referred as client side, is the visual appearance of a website in browser. The colour, font, buttons, menu, logo, images and all artistic arrangements of contents is the front-end of a website. HTML, CSS and JavaScript are the base languages a web developer uses to develop the front-end. HTML lays the web site contents in plain text. CSS organizes the contents, defines colour and fonts, arranges and places each components in right order making the website visually impressive. While HTML and CSS together construct a static webpage, JavaScript adds functionality to it.

Back-end or the server side refers to the behind-the-scene tasks of the website which users cannot see. Back-end programming basically deals with servers and databases. For example, when a user registers on a site, the provided information is stored in a database in a server. The user is unaware of what happens next after submitting his information in form. However, the back-end programming language responsively takes care of the data by storing it in the database and provides any requested service back from the server to the users. The behind-the-scene tasks are performed by various languages like PHP, python, ruby.

Any modern dynamic websites work closely between front-end and back-end. The front-end visually communicates with users while the back-end establishes communication between the server and the website. Figure-1 illustrates this process

Front-End vs Back-End

Front-end pieces the Web site owner can control: CMS software (WordPress, Drupal, Joomla), Web technologies (CSS, JavaScript, Flash)

Back-end elements are on the server (typically a Linux Apache MySQL PHP stack):

- software versions
- Web server modules (mod_gzip, mod_deflate, mod_SPDY, etc.)

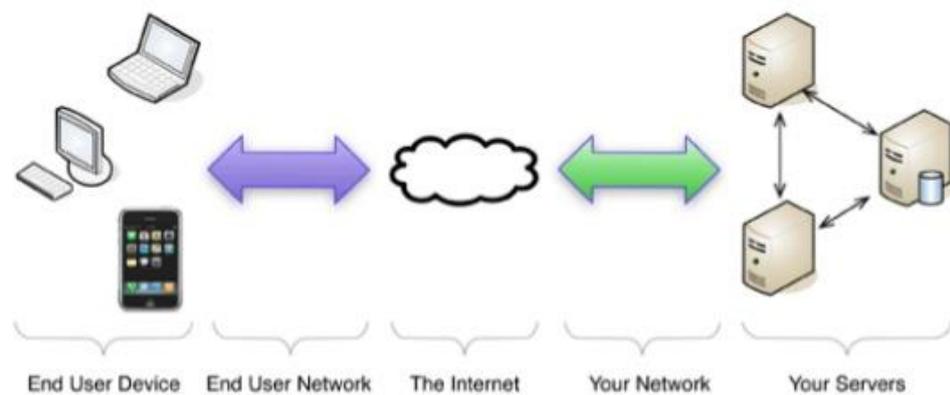


Figure 1. Front-End vs Back-End.

(Source: <http://www.slideshare.net/SEOGregKeith/making-web-sites-fast-greg-keith>)

3 MODERN WEB ELEMENTS

The requirements and designs of modern websites are different from what they used to be a few years back. Although their purpose has remained the same, the approach to presenting information and holding viewers' attention has changed considerably. Modern websites are creatively beautiful and architecturally complex. Therefore, the modern web development methods, tools and techniques have evolved to adhere to the needs of modern web designs. More new web elements have been introduced in order to achieve different effects. Few of such new elements which modern websites adopt are discussed below.

3.1 User Experience

User Experience, commonly referred as UX, is how a user perceives the look and feel of any products. The better the UX, the higher will be the profitability of products and number of satisfied customers. This principle applies to web development too. The success of a modern website depends upon how visitors perceive its overall look, design, functionality and usability [2].

A potential customer browses an organisation's website and learn about organisation. So the website has become an important interface between the organisation and customers, and hence, serves as a personality. Therefore, it is very vital for a website to look professional, attractive, and visually appealing. In addition, it should fully address the needs and requirements of a user and hold the interest of the user. It should be creatively presented and have well organised contents. Moreover, it should be simple, clean yet different from other websites.

So a web developer should consider many elements for a better UX. Some of such vital elements a web developer should always keep in focus when designing a web sites are discussed below.

3.1.1 Colour

Selecting an effective colour is very important for a website. The colour used should match the ambience of the website and communicate its message effectively. Colour choice should also concede its purpose and objectives.

The choice of colour depends upon the type of website we design, the products or services it advertises and the targeted audience. Nevertheless, the colour principle is same to all the websites. A website should have a balanced colour scheme. In other words a colour should be harmonically soothing in relation to the other colours. Similarly, colour contrast is equally important. A contrast in background and text colour helps in maintaining the readability. The use of bright and primary colours may be a good practise but should be used in right way. For example, red and orange are often used for indicating warning or errors while green and blue are used to attract the viewer's focus. Hence, the colours should be used meaningfully.

There are various tools available to assist developer choose a right colour schema for a website. Kuler, as seen in Figure 2, is an example of such colour palette which can be very handy.

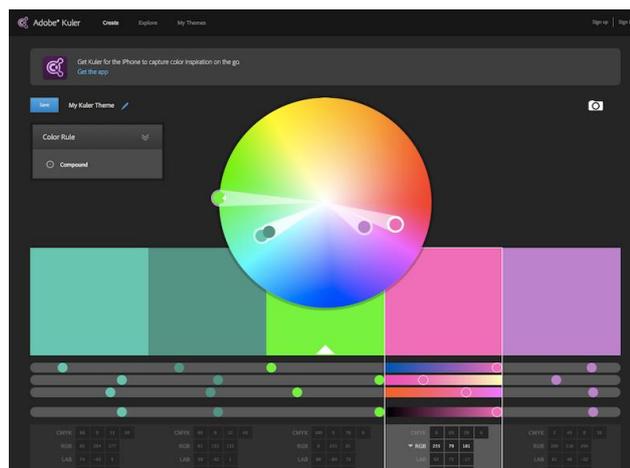


Figure 2. Kuler colour

(Source: <http://learntocodewith.me/posts/color-palette-tools/>)

3.1.2 Graphics

Graphics in a website refers to all the images, icons, and in the website. Whether it is a background image or company logo or save icon, graphics is a meaningful resource in a website. The use of graphics makes the contents concise without having to use too many textual contents. Since graphics is visual content, its uses definitely help in making a website interesting, attractive and intuitive. Moreover, creatively and accurately used graphics illustrates the information in more impressive way. For example, the use of a pie chart or bar graph to represent the demographic data helps in creating a good and long term impression on the viewers.

Graphics is a very common element in websites. It is hard to imagine a modern website without any graphics. While the use of graphics is a good practise, it also increases the size of the website which might result in longer loading time. So a developer should be extra cautious with the size of graphics so that the website loading time is not hindered.

3.1.3 Usability

Usability is an important commodity of UX. A user might have visited hundreds of websites and become used to modern websites architecture trends. So s/he is accustomed to finding the important components of sites, like menu, social media icons and contact information, in specific area of the page. Therefore a developer should design a website following this general convention so that users can easily navigate through the page. Further, it should be able to offer the information required by user with in a minimum number of clicks.

Clarity and consistency are important aspects of usability. A website should have a consistent colour, graphics, and designs, and should be visually clear and simple. Moreover, the information provided in a website is accessed by thousands of viewers. So the information should be true, reliable and well researched. Unreliable information can lead to mistrust of the site.

Similarly, the mobile support feature has been an imperative part of usability. Since the number of users browsing websites in mobile is increasing exponentially, it is important that the website supports in mobile browsers, too. This feature will help in keeping the number of users higher.

3.2 Carousel

It is often required for a website to accommodate influential information or advertise new products in its home page. In such circumstances, the carousel is very effective solution. It includes two or more high resolution images with few texts which slide at regular time intervals. The time interval is pragmatically set so that the sliding happens automatically. The number of images to include also depends upon the requirements. The image can cover full screen or just a small section. Since the website uses images and less text, it helps in visually illustrating the idea and promoting the products. Creatively placed carousel also makes the website alluring.

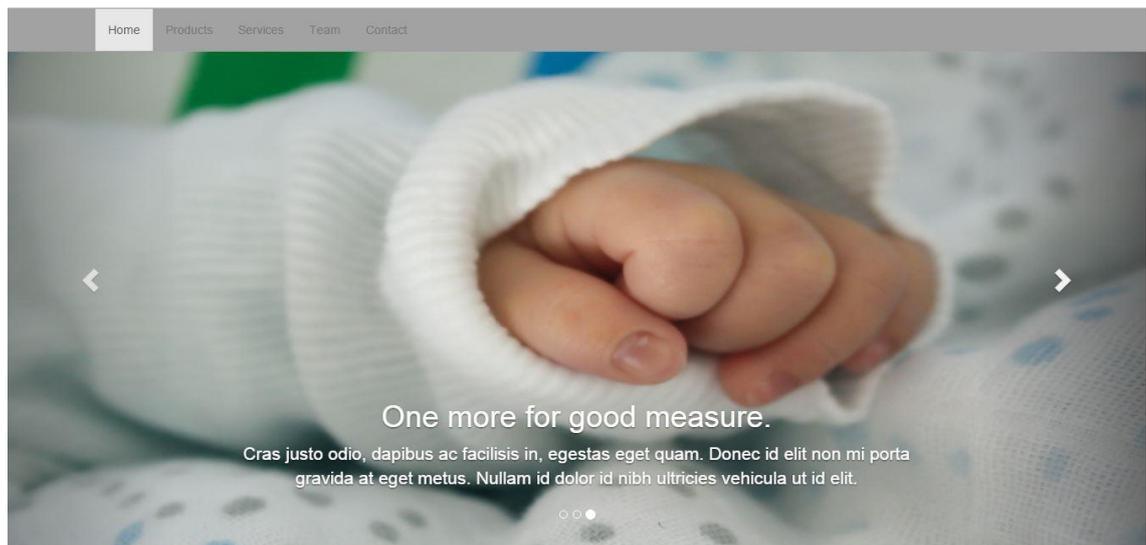


Figure 3. A Full page carousel.

Carousel effects can be achieved using widely available plugins. However, a simple carousel can be achieved with a few lines of code in JavaScript.

Table 1. Achieving a simple carousel in JavaScript.

```

var count=0;
window.setInterval(function slideshow() {
var arr=new Array();
arr[1]="images/baby3.png";
arr[2]="images/baby4.png";
arr[3]="images/baby5.png";
var img=document.getElementById("image_slider");

count=count+1;
if(count>5){count=1;}
if(count<1){count=5;}
img.src=arr[count];
},3000);

```

In the code above (Table 1), setInterval takes two parameters – slideshow function and the time. The slideshow function assigns the images in array. The time is set to be 3000 millisecond. So the image sliding interval for the carousel is 3 seconds.

3.3 Parallax scrolling

In the real world, an object at closer perspective seems to move faster than that at a farther perspective. This illusion is a phenomenon that exists because of 3D depth between the objects. This is referred to as the parallax effect.

The parallax effect has been in classic 2D and 3D games- but only recently it has been a trend in modern web development. The effect is achieved using multiple background layers laid relative to each other [3]. The layers at the front scrolls at greater speed than the layers at the back creating a faux 3D depth. So it creates a kind of illusion of 3D effect in the website. If creatively put, parallax effect scrolling can draw a simple story to keep viewers engaged. [Water nature](#) is an example of a powerfully used parallax scrolling effect in website.

Parallax scrolling may enhance user experience and users may find scrolling the website different than the conventional websites. However, it has to be deployed very cautiously. A developer should be aware of which type of website is suitable for this effect. Users' might get obsessed by too many of such effects. In addition,

if the internet connection is poor, it can further add the loading time and hence the user may completely avoid the site.

3.4 Scroll reveal

Scroll reveal is the method where web content reveals its position as users load a webpage or scroll through it. Instead of all the content being fixed into its position, it appears moving and adjusting into its position. For example, the title enters the viewport from left and reaches the centre as user loads the page. These types of effect do not necessarily add valuable information in the website but they certainly help in dragging user's focus to the important content of the site. User may have the impression that the webpage is interacting with him.

3.5 Transition, transform, and animation

In most of the modern websites, contents are not static. The contents are transitioning, transforming or animating in one or another way. These effects easily captivate users' attention and definitely enrich the elegance of the websites.

CSS3, the latest version of CSS, has introduced many new features that help in creating impressive transitions and animations. It is possible to change various CSS properties of HTML contents within a controlled time duration. Using CSS rules, we can determine the transition and transform properties, which element to animate, when to start it and for how long.

3.5.1 Transition

CSS transition allows a web developer to alter any property value over a controlled time duration. The change of the value is triggered by an event, such as mouse hover over or a click.

Table 2. Transition property in CSS.

```
#circle{  
  width: 150px;  
  height:150px;  
  background: red;  
  border-radius: 50%;  
  transition: background 2s;  
}  
#circle:hover{ background: green; }
```

In the piece of CSS above (Table 2), transition defines the change of background of the circle in 2 seconds. The triggering event is the mouse hover over it. So when the mouse is hovered over the circle, in two seconds, the back ground colour alters from red to green.

3.5.2 Transform

Scroll reveal is the method where web content is revealed into its position as user load a webpage or scroll through it. Instead of all the contents being fixed into its position, they appear moving and adjusting into their position. For example, the title enters the viewport from left and reaches the centre as the user loads the page. This type of effect does not necessarily add valuable information in the website but it certainly helps in dragging the user's focus to the important content of the site. The user may have an impression that the webpage is interacting with him.

CSS transform allows a web developer to alter shape, size and position of any elements. It supports 2D and 3D translation, rotation, scaling and skewing. Unlike the transition discussed above (3.5.1), it may not require an event to trigger the change. For example, a simple transform: rotate(100deg) function below rotates an element by 100 degrees (Table 3).

Table 3. Transform property in CSS.

```
#div {
transform : rotate(100deg);
}
```

3.5.3 Animation

Like transition and transform, animation also allows an alteration of a property value of HTML elements. However, the alteration of the value does not rely on any trigger events. The iteration of the animation may be finite or infinite. Additionally, animation can be both 2D and 3D. There are three major steps to achieve animation in CSS3 which are briefly illustrated below.

In CSS animation, keyframes hold the stack of elements' properties information at specific time. They are represented as `@keyframes` and bind necessary rules for the animation. `@keyframes` is followed by name of the animation (Table 4).

Table 4. Defining keyframes in CSS.

```
@keyframes flipCircle{
}
```

Keyframes rules define one or more properties of an element from start to the end of animation. The period of animation is denoted by *from* (start) and *to* (end). It can also be denoted by percentage (%) if there are more intermediate changes of properties value during the animation (Table 5).

Table 5. Defining transform property in keyframes.

```
@keyframes flipCircle{
0%{transform: rotateY(0deg) scale(1.25)}
25%{transform: rotateY(180deg) scale(1.25)}
50%{transform: rotateY(360deg) scale(1.5)}
100%{transform: rotateY(540deg) scale(1.25) }
}
```

After keyframes and the rule have been defined, keyframes have to be connected to the HTML elements (Table 6).

Table 6. Calling animation property inside the div element.

```
#circle{
  width: 150px;
  height:150px;
  background: blue;
  border-radius: 50%;
  animation: flipCircle 2s linear infinite;
}
```

In the code above (Table 6), animation properties define the name (flipCircle), duration (2 seconds), timing function (linear) and the iteration count (infinite).

The important consideration while using CSS animation is browser support. Although CSS3 is backward compatible to earlier versions, few of the old browsers do not support animation. So before implementing any animation, a developer should be cautious about the targeted browsers.

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
								4.1	
8			43					4.3	
9		40	44					4.4	
10	12	41	45	8	32	8.4		4.4.4	
11	13	42	46	9	33	9.1	8	44	46
	14	43	47		34				
		44	48		35				
		45	49						

Figure 4. A chart for CSS3 animation browser supports
(Source: <http://caniuse.com/>)

3.6 Responsiveness

With the advent of smartphones and tablets with various screen sizes, web designing has become more challenging than ever. The website designed for the 1024px desktop, certainly, does not optimize in the 480px iPhone or 680px Samsung tablet. Moreover, a website does not look quite the same in different device orientations (portrait and landscape). Since the users' device preference to surf a website is quite variant, it is the responsibility of a web designer to design and develop a website that optimizes in all the devices and orientations. So, to address this issue, the concept of responsive design was put forward. In fact, responsive design has long been used effectively in modern web design.

Responsive website refers to the ability of a website to automatically adapt and adjust to any devices, regardless of their screen dimension, orientation and resolution. The contents and layouts of a website arrange themselves to fit in the device screen size. So, what makes the website behave responsively? The answer lies in three features which are the backbone for crafting responsive websites – fluid layout, resizable images and media query [5].

3.6.1 Fluid layout

The layout and content width of the website is set to percentage (%) rather than fixed width (px). Therefore, the width of containers depends upon the viewport (screen) size. As a result, the width of the containers resizes while the browser size is adjusted when the website is viewed in various screen sizes.

Table 7. Defining the container's width in percentage.

```
#container{  
Width:60%;  
}
```

3.6.2 Resizable image

Similar to the layout, in responsive websites, it is important that images resize according to the viewport. So instead of defining a fixed height and width of image, they are defined in percentage. The important point to consider here is that the quality or resolution of the image should not be compromised while the images resize.

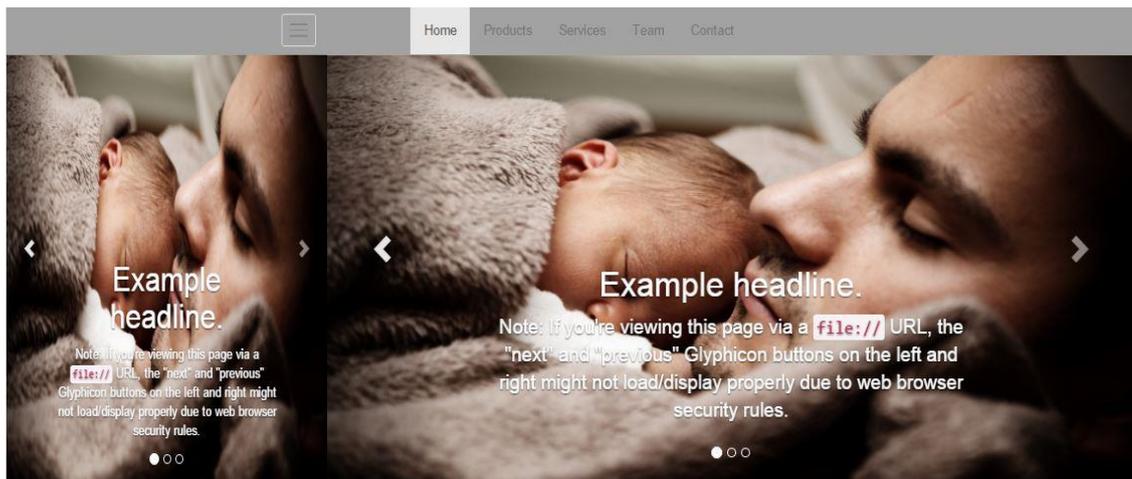


Figure 5. Mobile portrait view (left) and tablet landscape view (right).

3.6.3 CSS media query

Media query is the prominent feature of CSS3 (Cascading Style sheet) that helps in crafting a responsive website. By defining the media attribute (`@media`), it is possible to define the CSS rule for specific media types and devices [4]. There are various recognised media types, such as screen, print, projections, to name a few. The properties media attribute can define are the device's width, height, orientation, colour, aspect ratio, scan and grid. Two properties can also be defined at a time using logical operators.

Below is a common method to specify a media rule (Table 8).

Table 8. Meta tag in HTML.

```
<head>  
  <meta name="viewport" content="width=device-width, initial-scale=1.0">  
</head>
```

The meta tag is defined in the head section of the HTML page, as written above (Table 8). Keeping the initial scale to 1, the initial scaling or zooming of contents is set to default. The width=device-width sets the content of the page stretch to the total width of device.

Table 9. Media query defined in CSS.

```
@media scree and (min-device-width: 680px ) and (max-device-width:720px )  
{  
body {width: 60%;}  
}
```

In the example above (Table 9), @media refers to the initiation of the media rule inside CSS. The targeted media type is the devices' screen. The min-width and max-width defines the range of screen width (in pixels) in which the rule will be in effect. The curly braces enclose all the properties defined for this rule.

The fundamental principle behind responsive design is the ability of the media rule to target a specific device screen size. However, in practice, the end users' device preference may range from the 320px mobile screen size mobile to the 680px tablet to the 1920px wide screen desktop. So, is it necessary to create a media rule for each and every devices available? The answer is no. Even though there is a huge range of screen sizes, they can still be categorised. Below is the list of media rule that can commonly be used to cover the wide range of device screens (Table 10).

Table 10. Media rule in CSS.

```

//for mobile devices in landscape and portrait mode
@media screen and (max-width: 480px) {.....}
//landscape mobile mode to portrait tablet mode
@media (max-width: 767px) {.....}

//tablet to desktop
@media (min-width: 768px) and (max-width: 979px) {.....}
//larger desktop beyond 1200px
@media (min-width: 1200px) {.....}

```

Although these are the commonly used screen widths in media rule, there are not any recommended widths for responsive layouts. This totally depends upon the layout structures or the arrangements of contents in the page. Sometimes, it is rather important to take the breakpoint of the page's layout structures to define the screen width in media query than targeting a specific device width. For example, if arrangements of two floating divisions break at 817px, the minimum width definition in the query could be 817px.

3.6.4 Responsiveness in Bootstrap

Bootstrap is currently the most popular front end framework for developing websites and web applications [6]. This open source framework supports flexible layouts for creating responsive interfaces with less effort. It opts for the mobile first approach; that is the websites are designed for mobile and smaller devices and are extended to be supported in desktop. Since Bootstrap has built-in classes and media query for the default responsive feature, it has highly simplified the development of responsive websites.

The reason behind the Bootstrap being very effective in responsive web design lies in its grid system. As the Bootstrap's document on Grid System quotes [7]:

“Bootstrap includes a responsive, mobile first fluid grid system that appropriately scales up to 12 columns as the device or viewport size increases. It includes

predefined classes for easy layout options, as well as powerful mixins for generating more semantic layouts.”

As stated above, Bootstrap scales up to 12 columns in a container. It has four predefined classes for the container and each class is defined for the specific type of devices as described below.

col-xs-*

The extra small (xs) class is targeted for small screen devices, such as mobile phones which are less than 768px in width.

col-sm-*

The small screen class (sm) is for devices like tablets whose viewport is greater than 768px.

col-md-*

The medium (md) class targets desktop screens whose width is greater than 992px.

col-lg-*

The large (lg) class targets the larger screen desktop which are more than 1200px in width.

So, if a website is targeted at tablets, the grid system definition would be col-md-6 where six is the number of columns the container stretches across the device screen. Since, the grid system is fluid, the same configuration is transformed in mobile devices where the columns overlay to fit the mobile screen.

3.7 Multilingualism

In this era of globalisation, business no more remains local. A growing business company always tries to expand its market overseas or acquire more customers internationally. For any business to remain globally competitive, it has become vital to address the need of multinational and multicultural customers. Since a company website is the important interface, it is indispensable for any company to present information and services in its website in the language the customers understand.

Multilanguage website refers to the capability of website to support more than one language. As it is vital for a company to make a multilingual website or expand its existing website to a multilingual one, it is equally challenging for web developers to develop, maintain, and secure a multilingual website. Multilingualism in websites is achieved in various ways. The procedure applied in the geosmartcity (<http://geosmartcity.dc.turkuamk.fi/>) website to achieve it using PHP has been explained below.

File system

The Geosmartcity website requires to support three languages – English, Swedish, and Finnish. PHP, a backend scripting language, has been deployed for multilingual [8]. The project directory consists of a folder named “language” which contains the website contents in all three languages.

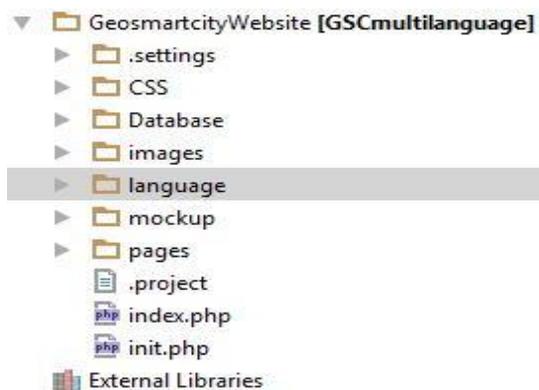


Figure 6: File structure of GeoSmartCity website.

The language folder consists of three sub-folders named en, fi, and sw. To keep them short, the sub-folder names are abbreviated for English, Finnish and Swedish languages respectively. Each of the sub-folders contains php files with contents in respective language (Figure 7).



Figure 7. Content of language directory

PHP session and GET request

The PHP Session is a temporary method to store valuable data that can be accessed from other pages of the website across a browser. It is written as `$_SESSION ['name']` where variable 'name' is the name of the session. This variable holds a piece of information which other pages can access. The session is erased once the browser is closed or by programmatically ending the session.

Whenever a user makes a GET request, the data is appended in the URL. The data is clearly visible in URL bar. In PHP, the appended data can be received by the `$_GET ['name']` function where 'name' holds a variable.

In the GeoSmartCity webpage, the PHP session is used to cache the page an end user is currently visiting and GET is used to determine the language the user has chosen. As soon as the user makes a request for the website, the `init.php` file is initiated. Since the user has not yet chosen the language and the contents, this file initiates the session and assigns a default language to Finnish and sets the default page to the home page.

Table 11. `init.php` file.

```
//starts the session
session_start();

//set Finnish as the default language
$_GET["language"]="fi";

//set default page to home page
$_SESSION["page"]="home";
```

Then the section of the webpage is loaded based on the `init.php` file (Table 12).

Table 12. Loading a page based on session.

```
if($_SESSION["page"]=="home"){
    include("language/".$_GET["language"]."/"."home.php");
}
```

The above piece of code (Table 12) sets the file route to language/fi/home.php

Since, “home” is assigned for the session, the include function looks for the file under the “language” folder which contains the sub-folder “fi”. The sub-folder contains a file named home.php, hence the home page in Finnish is loaded.

The user is now free to navigate to any pages in any language. User’s navigation is reflected in URL bar. For example if the user selects contact form menu and chooses Swedish, the information is appended in URL as - www.ge-smartcity.dc.turkuamk.fi/?page=contact&language=sw.

From the URL bar, the PHP GET function pulls the page and the language variable. The session is then assigned with new page variable (Table 13).

Table 13. GET and SESSION variables from URL.

```
$_GET["page"]=="contact" and $_GET["language"]=="sw"
    $_SESSION["page"]==$_GET["language"];
```

So the new route would be language/sw/contact.php which will display the contact.php in Swedish.

4 RESOURCEFUL WEB DEVELOPMENT

For a web developer, before starting to develop a website, it is crucial to understand the type and scale of the website a client is requesting. Resolving the tools to use and clinching the approach for the development beforehand not only saves considerable amount of time, but also results in a quality product. Additionally, to make a web development process more resourceful, a careful implementation of modern techniques is vital.

Basically, all the websites can be categorised into two types – static and dynamic. Since the functionality of these two types of websites is slightly different, the development approach is also different. The following sections illustrate different procedures and approaches for developing each type of website resourcefully.

4.1 Static websites

Static websites are the websites whose contents do not change. The web contents appear the same to all the users. Generally, they are simple and contain only few pages. The back end is written in plain HTML, CSS, and possibly JavaScript. Any updates or changes to the contents have to be manually done on the backend.

Although the static websites are simple and require less time to build, a developer has to consider many modern aspects like cross-browser compatibility, responsiveness and imposing appearance. Hence, starting to develop a website from scratch may surprisingly turn out time-consuming, tedious and error prone. So, to avoid this situation, it is always prudent to implement modern tools available. The best option would be to use a CMS (Content Management System) like WordPress, Drupal, Joomla and many more. Below are the reasons why CMSs are the best options for developing any type and scale of static websites.

4.1.1 Out-of-the box features

Most of the CMSs have inbuilt support for important features like responsiveness and browser compatibility. For example, WordPress has many responsive themes which are easily implemented. Further, the cross-browser compatibility is taken care of by default. Even without involving any backend coding, suitable and impressive web templates can be created, which saves a large portion of web development time. Standard wizards and different plugins can also be used for adding various elements in the page.

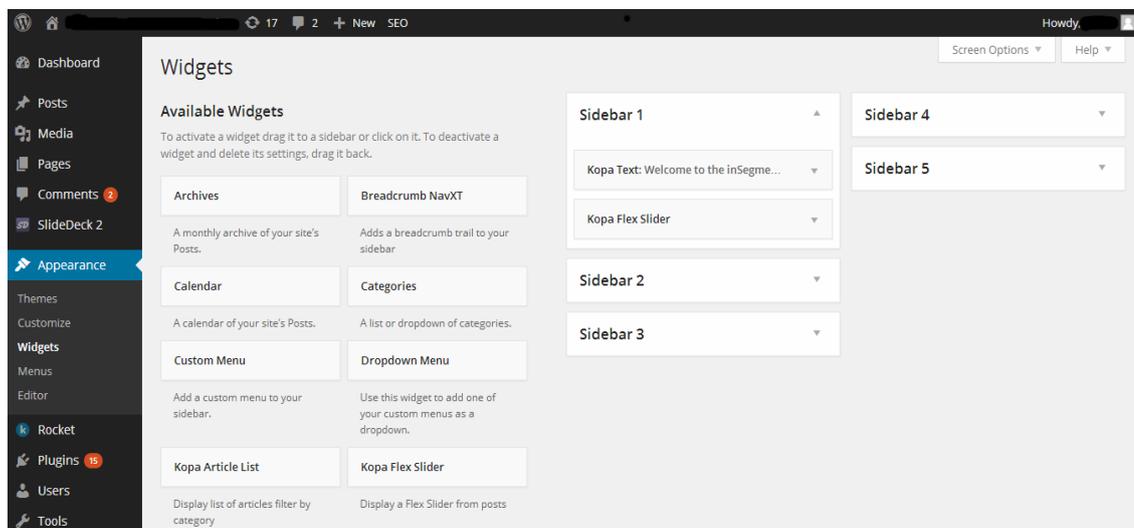


Figure 8. WordPress dashboard for adding widgets.

(Source: <https://www.wikipedia.org><https://www.wikipedia.org/>)

4.1.2 Maintenance

Normally, if updates are to be made in a static website, a developer has to alter the source files. This might require great precautions and more time. If the website is large, altering many files is tedious and prone to encounter problems. However, CMSs have addressed these issues. The update can be made easily just by altering the contents, rarely requiring to change the source files. So, using CMSs not only saves time, but also ensures an effective update.

4.1.3 Extensibility

It is most likely that a website's requirements grow by time. Static websites might have to be expanded to contain a few or more dynamic contents on the page. For example, a company wants to add a feedback form or include social networking links on its static website. In such cases, CMSs painlessly help in extending such features. Hundreds of free plugins and inbuilt features assist in any minor extensibility of the websites.

4.2 Dynamic websites

Dynamic websites are the websites whose contents change depending upon the action and preferences of users. The change might be automatic or as a result of user's interaction with the website. The dynamism of the website comes from the server-side with the combination of the client-side scripting in the backend. Since a developer has to write a server side scripting language, the web development process turns fairly complex and consumes much time if the website is big.

A CMS, like in the case of static website, can be used for basic and simple type of dynamic websites. For example, if a website requires to handle user registration or include Facebook likes and comments on website posts, CMS like WordPress can be used. Since a CMS can handle simple database, basic dynamic features can be managed in the CMS, too. Nevertheless, in most of the cases, dynamic websites have more than such basic requirements. They have a fairly complex database system and the required backend support is huge. In such case, the web development exceeds the content management. So, a more powerful tool or method is required. As in the case of static website discussed earlier, if a developer wants to avoid developing a dynamic website from scratch, they can acquire web framework.

“A web framework (WF) or web application framework (WAF) is a software framework that is designed to support the development of web applications including web services, web resources and web APIs. Web frameworks aim to alleviate the overhead associated with common activities performed in web development. [9]”

All the dynamic websites have some common functionalities, such as database queries, session handling, user authentication, data validations etc. So as explained above, a web framework facilitates web development process by providing inbuilt and ready to use common tasks so that a developer does not have to rewrite code each time. This arguably saves a huge amount of time and undoubtedly eases the developer's work.

Web frameworks are gaining popularity and more and more developers are adopting them. Developers can choose a suitable and robust web framework from a wide range of web frameworks on the Internet. To illustrate why web framework is preferably the best option for effective web development, the Laravel framework is briefly explained in the following section.

4.2.1 Laravel - a PHP framework.

Laravel is one of the most popular PHP frameworks on the Internet [10]. It is open source and as most of the web frameworks, it follows MVC (Model-View-Controller) architecture [11]. The MVC architecture separates logical data structure from user interface and promotes code reusability.

Model

The Model is an object that represents the logical data structure. It is a database class that serves the database records to the controller classes on request. In Laravel, the model represents one table in a database.

Controller

The controller represents bridging classes between views and Model. It listens the request from users in the Views and executes the requests by getting necessary data from the Model class. The data is then passed back to views and displayed to the user. Thus, it acts as a mediator between views and the model.

View

The View is, basically, data and information presented in the User Interface (UI). The data to be displayed to user depends upon the Controller classes and their

functions. User fires a request or performs an action in the View which triggers Controller classes to execute the request.

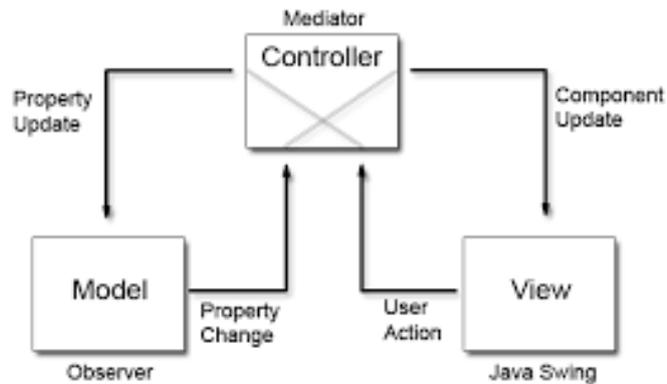


Figure 9. A core relation between Model, Controller and View.

(Source: <http://www.codeproject.com/Articles/25057/Simple-Example-of-MVC-Model-View-Controller-Design>)

The RESTful routing

The most commonly executed task in database is create, read, update and delete, which is often abbreviated as CRUD. During the backend development, many functions and classes have to be written and re-declared to perform these tasks. Hence, Laravel has made this common task easier by introducing the RESTful resource controller [12]. Let us take the domain for our sample website to be www.example.com.

The RESTful controller for the [example.com](http://www.example.com) is defined as in the table 14.

Table 14. Defining the resource controller in Laravel.

```
Route::resource('/user', 'ManageUser');
```

The resource declaration creates all the necessary routes for 'user'. So whenever the URL receives the request for user (example.com/user), it points to the ManageUser controller class which contains all the necessary CRUD functions for GET, POST and PATCH/PUT http requests. Table 15 lists the routes that are auto generated by Laravel.

Table 15. List of routes generated by RESTful routing.

HTTP	PATH	ACTION/METHOD	DESCRIPTION
GET	example.com/user	index	Fetches users from database and return to view for display
GET	example.com/user/create	create	Returns the view for creating/registering a new user.
POST	example.com/user	store	Catches the input when user requests to register and save it to database.
GET	example.com/user/{id}	show	Fetches the user from database whose id is {id} and return to view
GET	example.com/user/{id}/edit	edit	Shows the form in view where user can edit the information
PUT/PATCH	example.com/user/{id}	update	Updates the user information whose id is {id}
DELETE	example.com/user/{id}	delete	Deletes the user from database whose id is {id}

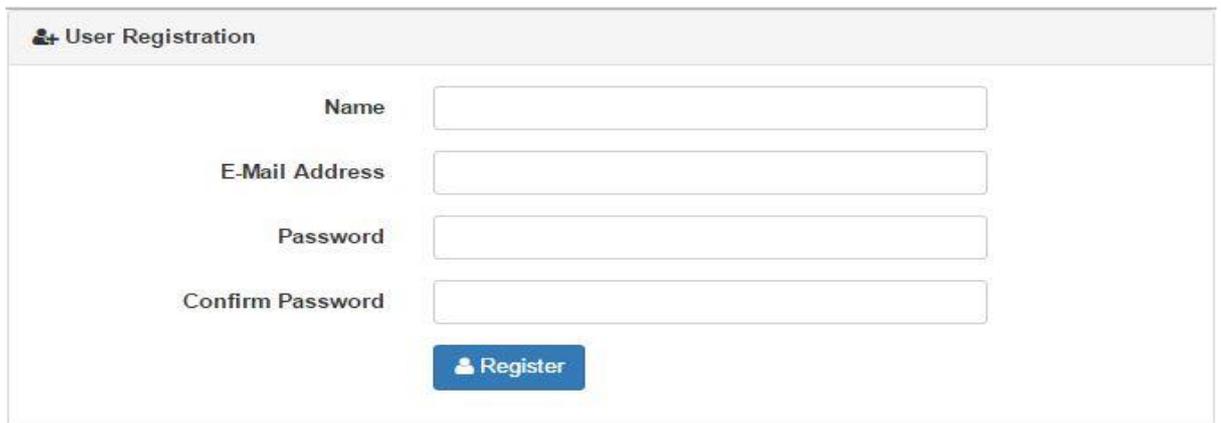
Since all the routes and their related functions are generated, it becomes easier to access Models and fetch records to the Views. The database records can be painlessly accessed using the Laravel Model classes in the Controller, never requiring to write complex database queries.

When example.com/user/create is requested in the URL, create function is triggered in the ManageUser Controller.

Table 16. Display the register View from Controller.

```
public function create()
{
    return view('register.register');
}
```

The create function, as listed in the table 16, is responsible for displaying the form where the user can register. So, in this example, the function redirects to the View. The View then calls for a class named register inside the directory register. The register class, which contains HTML form elements, displays the registration form to the user.



The screenshot shows a web form titled "User Registration". It features four text input fields stacked vertically, each with a label to its left: "Name", "E-Mail Address", "Password", and "Confirm Password". Below these fields is a blue button with a white user icon and the text "Register".

Figure 10. Registration form displayed by View.

When a user fills the form and submits, a POST request is triggered. Then the store function in the Controller is called which will save the data to the database table. However, before saving the data, it is very important to validate it so that saving inappropriate data in database is avoided. The validation and saving of the data is executed in the store function.

Table 17. User data is validated and stored from the Controller.

```
public function store(Request $request)
{
    $validation=Validator::make($request->all(),[
        'name'=>'required|min:4',
```

```

        'email'=>'required|max:255|unique:users',
        'password'=>'required|max:255'

    ]);
    if($validation->fails()){
        return "Error in information you provided ! Please provide all the information correctly";
    } else {
        User::create([
            'name'=>$request->name,
            'email'=>$request->email,
            'password'=>bcrypt($request->password)
        ]);
        return "You have been registered";
    }
}

```

In the store function, the make function of the class Validator accepts two parameters. The first parameter (`$request->all()`) holds all the data submitted by the user in the form. The second parameter is the validation rule. In the case above (table 17), the name is validated against two rules: the value cannot be empty and the minimum character should be four.

In the next step, if the validation fails, the user is informed about it and asked to provide the correct information. However, if the validation passes, the data is forwarded to the next step.

The User model is the Model class for the database table 'user'. Behind the scenes, it contains all the necessary functions to conduct CRUD actions to the user table. Laravel has taken care of all those behind-the-scene functions for web developers who just need to declare the Model class and call its instance wherever and whenever required.

Table 18. User Model class.

```
public class user()
{
    //user model
}
```

The few lines of class declaration code (table 18) is powerful enough to handle all the actions web developers require. In the store function, the user model and its create function is instantiated – `User::create()`. In the create function, the user's input is assigned to the respective column of the user table. This completes the process of creating a new user.

Accordingly, accessing the records from the database is made easy in Laravel. When the `example.com/user` URL is requested, the index function in the `ManageUser Controller` is triggered.

Table 19. User's data is passed from the Controller to the View.

```
public function index()
{
    Return view('display.users')->with('user',User::all());
}
```

Since the index function is responsible for displaying the user's information, it has to fetch the data from the database table and pass it to the View. Thus, the function declared above returns 'users' view which is in the display directory. `User::all()` calls the User Model and gets all the data from the user table as an array. The data is then attached to the key word 'with'. Hence, the function points to the users view and passes all the data from the users table to the view.

Table 20. Users View page.

```

@extends('master')
@section('content')
    <table class="table table-bordered">
        <thead>
            <tr>
                <th>Id</th><th>Name</th><th>Email</th>
            </tr>
        </thead>
        <tbody>
            @foreach($user as $user)
                <tr>
                    <td>{!! $user->id !!}</td><td>{!! $user->name !!}</td><td> {!! $user-
                >email !!}</td>
                </tr>
            @endforeach
        </tbody>
    </table>
@endsection

```

In the users View (table 20), the data passed from the Controller is accessed as the PHP variable `$user` and displayed in the table. Finally, the output of the `example.com/user` URL request will be as in the figure 10.

Id	Name	Email
1	john carlos	johncarlos@gmail.com
2	John doe	doe.john@gmail.com
3	kerry petty	kerrypetty@hotmail.com
4	Michael sennata	sennatta@yahoo.com

Figure 11. Output of the URL `example.com/user`

This simple example illustrates how powerful the Laravel web framework is. All the functions and classes have already been declared behind the scene. Data from any tables in database can be acquired without even writing a single line of database query. The code is clear and easy to write which simplifies and makes a web developer's task much faster. The framework can be so reliable and effec-

tive that the developer does not have to be concerned with what is going on behind the classes and functions that s/he is using. Moreover, hundreds of ready-to-use packages and libraries make the framework better choice than traditional methods of web development.

In fact, there are plenty of web frameworks available. All the frameworks, more or less, work in a similar fashion as explained above. The syntax and features might vary, but every web framework intends to simplify the developer's work. When developing a business and corporate websites, the required features might be huge and complex. Developing such websites from scratch consumes a massive amount of time and hence become uneconomical. Thus, using a popular and reliable framework makes the web development highly economical.

5 CONCLUSION

Web development trend and technology are constantly changing. The appearance of modern websites and the way information is presented are far superior and out-reaching than before. Since the required services and features in modern websites are huge, the modern web development tools and techniques are rapidly advancing to meet these requirements.

Similarly, variant web elements have been introduced in modern web development to enhance the look and feel of websites. Colour and graphics are profoundly used to express and emphasize the meaning of the web contents. Images are artistically presented in various forms, such as carousel and parallel scrolling. In addition, animations, transition and transform have become common and important parts of modern web elements. While these kinds of elements are keys to designing modern websites, they add complexity to the development process. As a result developer should be well aware of all such requirements and should be able to implement them whenever required.

Currently, a back end user has abundant options to browse websites. The user can access websites in devices ranging from a 320px mobile phone to a1920px desktop. The wide range of device screens have urged the necessity of responsiveness in websites. To develop a responsive website, CSS rules can be defined in Media query targeting specific screen width. A more effective way to achieve responsiveness would be to use the Bootstrap framework. Bootstrap's mobile first approach provides flexible layout which adjusts to any device screen.

Additionally, multilingualism is an approach to reach multicultural and multinational users. To expand a business globally, companies should advertise their products and services in the language the customers understand. With the PHP session and the GET method, a simple file system can be deployed to achieve multilingualism in websites.

Moreover, one of the most important aspects of developing a modern website is resourceful development. Modern organisational or business websites have a

huge requirements. Developing such websites in a traditional way requires a huge amount of time and resources. So, to overcome these issues, developers can opt for available modern web development tools. For simple static websites, a CMS, such as WordPress can be used. However, for complex dynamic websites, a web framework like Laravel can be used. Many built-in features and ready-to-use functionalities of CMSs and web Frameworks make the web development process faster, effective and more profitable.

REFERENCES

- [1] Kalie Kavourgias. What is the Difference between Front-End and Back-End, [Online] Available From: <http://www.bitrepository.com/php-how-to-add-multi-language-support-to-a-website.html> [Accessed: 10th September 2015].
- [2] Vitaly Friedman and Sven Lennartz. What is User Experience Design? Overview, tools and resources, [Online] Available From: <https://www.smashingmagazine.com/2010/10/what-is-user-experience-design-overview-tools-and-resources/> [Accessed: 2nd October 2015].
- [3] WikiPedia. Parallax Scrolling, [Online] Available From: https://en.wikipedia.org/wiki/Parallax_scrolling [Accessed: 15th September 2015].
- [4] MDN. Using media queries, [Online] Available From: https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries [Accessed: 15th November 2015].
- [5] Kyla Knight. Responsive web design: what is it and how to use it, [Online] Available From: <https://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/> [Accessed: 7th November 2015].
- [6] Syed Fazle Rahman. Responsive Web Design Tips from Bootstrap css, [Online] Available from: <http://www.sitepoint.com/responsive-web-design-tips-bootstrap-css> <http://www.sitepoint.com/responsive-web-design-tips-bootstrap-css/> [Accessed: 24th September 2015].
- [7] BootstrapDocs v3.3.6. Grid System, [Online] Available From: <https://bootstrapdocs.com/v3.3.6/docs/css/> [Accessed: 28th September 2015].
- [8] Gabriel Livan. How to add PHP Multilanguage Support to Website, [Online] Available From: <http://www.bitrepository.com/php-how-to-add-multi-language-support-to-a-website.html> [Accessed: 22nd September 2015].

[9] Wikipedia. Web Framework, [Online] Available From: https://en.wikipedia.org/wiki/Web_framework [Accessed: 4th October 2015].

[10] Laravel 5.2. Laravel Web Framework, [Online] Available From: https://en.wikipedia.org/wiki/Web_framework [Accessed: 10th October 2015].

[11] Jeff Atwood. Understanding Model-View-Controller, [Online] Available From: <http://blog.codinghorror.com/understanding-model-view-controller/> [Accessed: 4th October 2015].

[12] Chris Sevilleja. Simple Laravel CRUD with Resource Controller, [Online] Available From: <https://scotch.io/tutorials/simple-laravel-crud-with-resource-controllers> [Accessed: 10th September 2015].