|  |
| --- |
|  |
| **DEPARTMENT OF COMPUTER ENGINEERING** |
| **CSL504 Web Design Lab** |
| **Fifth Semester, 2020-2021 (Odd Semester)** |

**Name of Student   : Ambar Jha**

**Roll No.                  : 18**

**Division                  : TE- CMPN**

**Assignment No     : 4**

**Outcome           :  CSL504.5**- Understand the basics of XML, DTD and XSL and develop web pages using XML / XSLT

**CSL504.6**  Analyze end user requirements and Create web application using  appropriate web technologies and web development framework

**Task                          :  XML**

**Date of Assignment   :**

**Date of Submission   :**

|  |  |  |
| --- | --- | --- |
| **Particulars** | **Max. Marks** | **Marks Obtained** |
| Timely Submission (TS) | **3** |  |
| Originality of material (OM) | **3** |  |
| Neatness (NT) | **3** |  |
| Innovative Solution (IS) | **3** |  |
| **Total** | **12** |  |

**Grades – Meet Expectations (3 Marks), Moderate Expectations (2 Marks), Below Expectations (1 Mark)**

**Checked and Verified by**

**Name of Faculty    :  Pravin Jangid**

**Signature                :**

**Date                        :**

Assignment No. 4

Sub: **Web Design Lab**               Sem: **V** Branch: **CS**

**1. What is a DTD? Why Use a DTD? Write XML document with an internal DTD**

**Ans:**

A DTD is a Document Type Definition.

A DTD defines the structure and the legal elements and attributes of an XML document.

Why Use a DTD?

With a DTD, independent groups of people can agree on a standard DTD for interchanging data.

An application can use a DTD to verify that XML data is valid.

An Internal DTD Declaration

If the DTD is declared inside the XML file, it must be wrapped inside the <!DOCTYPE> definition:

**XML document with an internal DTD**

<?xml version="1.0"?>

<!DOCTYPE note [

<!ELEMENT note (to,from,heading,body)>

<!ELEMENT to (#PCDATA)>

<!ELEMENT from (#PCDATA)>

<!ELEMENT heading (#PCDATA)>

<!ELEMENT body (#PCDATA)>

]>

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend</body>

</note>

The DTD above is interpreted like this:

* !DOCTYPE note defines that the root element of this document is note
* !ELEMENT note defines that the note element must contain four elements: "to,from,heading,body"
* !ELEMENT to defines the to element to be of type "#PCDATA"
* !ELEMENT from defines the from element to be of type "#PCDATA"
* !ELEMENT heading defines the heading element to be of type "#PCDATA"
* !ELEMENT body defines the body element to be of type "#PCDATA"

**2. What is an XML Schema (XSD).Explain it with e.g.**

**Ans:**

An XML Schema describes the structure of an XML document.

The XML Schema language is also referred to as XML Schema Definition (XSD).

XSD-Example:

**<?xml version="1.0"?>**

**<xs:schemaxmlns:xs="http://www.w3.org/2001/XMLSchema">**

**<xs:element name="note">**

**<xs:complexType>**

**<xs:sequence>**

**<xs:element name="to" type="xs:string"/>**

**<xs:element name="from" type="xs:string"/>**

**<xs:element name="heading" type="xs:string"/>**

**<xs:element name="body" type="xs:string"/>**

**</xs:sequence>**

**</xs:complexType>**

**</xs:element>**

**</xs:schema>**

The purpose of an XML Schema is to define the legal building blocks of an XML document:

* the elements and attributes that can appear in a document
* the number of (and order of) child elements
* data types for elements and attributes
* default and fixed values for elements and attributes

XML Schemas Support Data Types

One of the greatest strengths of XML Schemas is the support for data types.

* It is easier to describe allowable document content
* It is easier to validate the correctness of data
* It is easier to define data facets (restrictions on data)
* It is easier to define data patterns (data formats)
* It is easier to convert data between different data types

XML Schemas use XML Syntax

Another great strength about XML Schemas is that they are written in XML.

* You don't have to learn a new language
* You can use your XML editor to edit your Schema files
* You can use your XML parser to parse your Schema files
* You can manipulate your Schema with the XML DOM
* You can transform your Schema with XSLT

XML Schemas are extensible, because they are written in XML.

With an extensible Schema definition you can:

* Reuse your Schema in other Schemas
* Create your own data types derived from the standard types
* Reference multiple schemas in the same document

XML Schemas Secure Data Communication

When sending data from a sender to a receiver, it is essential that both parts have the same "expectations" about the content.

With XML Schemas, the sender can describe the data in a way that the receiver will understand.

A date like: "03-11-2004" will, in some countries, be interpreted as 3.November and in other countries as 11.March.

However, an XML element with a data type like this:

<date type="date">2004-03-11</date>

ensures a mutual understanding of the content, because the XML data type "date" requires the format "YYYY-MM-DD".

**3. Explain the MVC Architecture.**

**Ans:**

(1) A model that defines the elements of a web application and how they

interact.

View - Produces output

Model - Handles data

Controller - Orchestration/Routing

(2) What it really does is, it breaks the request response cycle at least how

we can handle it into three basic operations and it's helpful to have this

terminology so I can say, "Oh, that's the model that's the view and this

other things the controller."

(3) The basic definition in a pure sense is that we take data, we might update

the model(models are permanent storage), the controller is this thing that

decides when to do things and when to switch from one screen to another. and

the view is the next thing that we see. we could think of it as it comes

into the controller, the controller updates the model, the model then is done

maybe when you read the data which is reading from the model, we produce a view

and then the user is involved.

(4) There turns out to be lots of lots of different ways inside of code and if

you're familiar with or if you heard of things like CakePHP or Angular or

React or Symphony, These are all ways of interpreting the Model-View-Controller

differently.

(5) If you look in a framework they will put sort of Model stuff in one file,

 the controller stuff in another file and the view stuff in another file.

And then they will knit those things together. When we learned that there is

no right or wrong, there are just patterns that help organize things and so

if you're going to use, or do a symphony app and you've done a bunch of

symphony apps you know where all the models are at, You know where all the

views are at, you know where all controllers are at.

**4. Explain Laravel Framework.**

**Ans:**

Laravel is an open-source PHP framework, which is robust and easy to understand. It follows a model-view-controller design pattern. Laravel reuses the existing components of different frameworks which helps in creating a web application. The web application thus designed is more structured and pragmatic.

Laravel offers a rich set of functionalities which incorporates the basic features of PHP frameworks like CodeIgniter, Yii and other programming languages like Ruby on Rails. Laravel has a very rich set of features which will boost the speed of web development.

If you are familiar with Core PHP and Advanced PHP, Laravel will make your task easier. It saves a lot time if you are planning to develop a website from scratch. Moreover, a website built in Laravel is secure and prevents several web attacks.

**Advantages of Laravel**

Laravel offers you the following advantages, when you are designing a web application based on it −

* The web application becomes more scalable, owing to the Laravel framework.
* Considerable time is saved in designing the web application, since Laravel reuses the components from other framework in developing web application.
* It includes namespaces and interfaces, thus helps to organize and manage resources.

Composer

Composer is a tool which includes all the dependencies and libraries. It allows a user to create a project with respect to the mentioned framework (for example, those used in Laravel installation). Third party libraries can be installed easily with help of composer.

All the dependencies are noted in composer.json file which is placed in the source folder.

Artisan

Command line interface used in Laravel is called Artisan. It includes a set of commands which assists in building a web application. These commands are incorporated from Symphony framework, resulting in add-on features in Laravel 5.1 (latest version of Laravel).

**Features of Laravel**

Laravel offers the following key features which makes it an ideal choice for designing web applications −

1) Modularity

Laravel provides 20 built in libraries and modules which helps in enhancement of the application. Every module is integrated with Composer dependency manager which eases updates.

2) Testability

Laravel includes features and helpers which helps in testing through various test cases. This feature helps in maintaining the code as per the requirements.

3) Routing

Laravel provides a flexible approach to the user to define routes in the web application. Routing helps to scale the application in a better way and increases its performance.

4) Configuration Management

A web application designed in Laravel will be running on different environments, which means that there will be a constant change in its configuration. Laravel provides a consistent approach to handle the configuration in an efficient way.

5) Query Builder and ORM

Laravel incorporates a query builder which helps in querying databases using various simple chain methods. It provides ORM (Object Relational Mapper) and ActiveRecord implementation called Eloquent.

6) Schema Builder

Schema Builder maintains the database definitions and schema in PHP code. It also maintains a track of changes with respect to database migrations.

7) Template Engine

Laravel uses the Blade Template engine, a lightweight template language used to design hierarchical blocks and layouts with predefined blocks that include dynamic content.

8) E-mail

Laravel includes a mail class which helps in sending mail with rich content and attachments from the web application.

9) Authentication

User authentication is a common feature in web applications. Laravel eases designing authentication as it includes features such as register, forgot password and send password reminders.

10) Redis

Laravel uses Redis to connect to an existing session and general-purpose cache. Redis interacts with session directly.

11) Queues

Laravel includes queue services like emailing large number of users or a specified Cron job. These queues help in completing tasks in an easier manner without waiting for the previous task to be completed.