Item -1:

* Introduce to all
* Understanding web, website, and internet.
* Understanding how web works
* Technology for web structure, content & presentation.
* HTML history.
* Basic HTML-5 structure.
* HTML tag, element & Attribute.

Item -2:

* HTML Table Element
* HTML-5 text formatting element
* HTML-5 new element.
* Step by step static site using only HTML
* GIT

Item -3:

* CSS history
* CSS Basic Structure
* Layout with grid
* Basic navigation menu.

Item -4:

* Introduction to bootstrap -4.
* Bootstrap -4 Grid Structure.
* Bootstrap -4 typography, table, images.
* Bootstrap -4 Jumbotron, Form & Input Element.

Item -5:

* Bootstrap -4 Navbar.
* Bootstrap -4 Cards.
* Bootstrap -4 Accordion.
* Bootstrap -4 Carousel.
* Bootstrap -4 Modal.

Item -6:

* Understanding the behavior of web.
* JavaScript history & basic Syntax.
* Introduction to JavaScript event handling.
* JavaScript variable, Data Type, Operator & Statement.

Item -7:

* Working with JavaScript Array & Function
* Working with DOM.
* Object & Prototype in JavaScript.

Item -8:

* What is JQuery?
* JQuery basic syntax.
* JQuery Different Type of Selector

Item -9:

* Complete form validation using JQuery.
* JQuery image gallery
* JQuery show hide password
* JQuery scroll event

Item -10:

* Understanding server side technology.
* PHP history.
* PHP development environment.
* PHP Basic Syntax.
* PHP Variable & Data Type.
* PHP Function

Item -11:

* Understanding the concept of 00P
* Introduce to Class
* Introduce to Property
* Introduce to Method.
* Introduce to Object
* PHP Operator

Item -12:

* PHP Statement.
* PHP Array
* PHP super global Array.
* Basic input & output.

Item -13:

* Programming with object oriented PHP.
* Popular String related Function.
* Popular Array related Function.
* Understanding the concept of Inheritance.
* Understanding the concept of Encapsulation.

Item-14:

* Introduce to magic method.
* Introduce to Composer.
* Introduce to Autoload
* Introduce to Namespace
* Introduce to Trait.

Item- 15:

* Introduce to Database Server.
* Introduce to MySQL Database Server.
* Create database, table inside the Database Server.
* Introduction to Structured Query Language (SQL).
* Perform action through SQL inside Database Server.

Item-16:

* Introduction to CRUD operation.
* Data Save into the database server.
* Data Retrieve form the Database Server.

Item-17:

* Data edit & update.
* Data Delete.

Item-18:

* Understand the Session
* Understand the Login Concept.
* Create secure Login Panel.
* Prepare admin panel for A Blog Site. .

Item-19:

* Prepare Front End for A Blog A Site.
* Prepare category wise blog.
* Prepare Blog Details.

Item-20:

* Understanding the concept of framework.
* History of PHP framework.
* Popular PHP framework.
* Why Laravel.
* Introduce to MVC
* Laravel installation.

Item-21:

* Project folder structure.
* How Home Page Load.
* Route Introduction & Route Naming.
* Load String from Route for specific request.
* Load Array from Route for specific request.
* Load View from Route for specific request.

Item-22:

* Request Lifecycle
* Resource view route.
* Artisan command & Create Controller Class.
* Load Controller Class from Route for specific request.
* Pass data from Controller Class to view by Compact.
* Pass data from Controller Class to view by Array.
* Pass data from Controller Class to view by with.

Item-23:

* Introduce to Blade Template Engine.
* Introduce to directive Blade Template Engine.
* Laravel Blade Templating.
* Create Blade sub view.

Item-24:

* Introduce to Package in Laravel.
* Introduce to Laravel Collective.
* Install Laravel Collective Package.
* Create form & input element through Collective Package.
* Introduce to Migration, Schema, and Blueprint.

Item-25:

* Create migration file.
* Prepare migration file for creating table.
* Create & update table through command.
* Introduce to Factory, Faker & Seeding.
* Working with Factory, Faker & Seeding.

Item-26:

* Introduce to Quire builder in Laravel.
* Introduce to Eloquent ORM in Laravel.
* Create Model Class with migration file in one command.
* Data insert in the database through Quire builder.
* Data insert in the database through Eloquent ORM.
* Data Retrieve from database through Eloquent ORM & Query Builder.

Item-27:

* Data Edit & Update through Eloquent ORM & Query Builder.
* Data Delete through Eloquent ORM & Query Builder.

Item-28:

* Introduce to Auth Class.
* User registration & Login through Auth Class
* HTTP Basic Authentication.
* Introduce to middleware.
* Create, Register & Use Middleware.
* Create authentication using route group.

Item-29:

* Introduce to App Service Provider Class.
* Share data with specific view through view composer.
* Working with Mail.
* Working with PDF.
* Working with Notification & Pagination.

Item-30: (Project Work - Ecommerce Site)

* Discuss about SDLC.
* Select front Template.
* Select Admin Template.
* Front End Mastering.

Item-31:

* Admin Panel mastering.
* Manage admin login & register with security.
* Working with Category & Brands.
* Working with Products.

Item-32:

* Manage Home Page Content.
* Manage Category Page Content.
* Manage Product Details Page Content.

Item-33:

* Implement Shopping Cart.
* Customer Login & Registration for Checkout.
* Implement Ajax in Laravel.
* Manage Product Shipping Info.

Item-34:

* Manage Payment Info & Save the Order.
* Manage Order Form Admin.

Item-35:

* Understanding front end framework.
* What & Why Vue.
* Environment for Vue JS.
* Install Node JS & Vue CLI.
* Install Vue JS.
* Project run & home page.
* Project folder structure.
* Understanding the concept of component.
* Vue Single File Component
* Communication Between Vue & Laravel

Item-36:

* Project Presentation.

1st Class

* **Web** - Net/Jal - A **System**.
  + Many Document connect each other.
  + 1952 - Scientist Team share their knowledge each other using Electronics Computer.
  + Sir Tim Berners in 1989 create WEB.
  + How WEB Works ?
    - Client: Who’s do Request Processing & Request Submit with a Request machine.
    - Server - local & live.
* **Internet** 
  + In 1990 Internet created. First Internet names ‘**Packet Networking**’. US Army create first. Every Domain controlled by them.
  + Any one access any document through internet.
* **WebSite - A Document.** Information share on a particular category.
  + A complete documentation provide on a particular Subject.
  + Exp: bitm.org.bd.
* **WebPage - A Specific Document.** Information share on a particular topic.
  + A part of Website. Exp: bitm.org.bd/index.html
* **Web Application** means **Software  
  - 2 types software - Desktop, Web Based.  
  - has 3 parts**
  + FrontEnd
    - Structure & Content = HTML
    - Presentation = CSS
    - Behaviour = JS (Communication Between Web Content & Client)
      * ECMAScript – JS Version
  + ServerSite
    - PHP - v-7/ JAVA/ Python/
      * Laravel – e-commerce, OOP –Blog Site
      * Procedural language – Procedural PHP, OOP PHP
      * Structured language – Java, C#
  + Database
    - MySQL
* **Tools** - PHPStorm(JetBrains), SubLime Text, WEB Strome
  + License Server: <http://idea.imsxm.com> , Discover -> Activate
  + http://kadara.ru:1017
  + http://roothat.ru:1017
  + http://jetbrains.tencent.click
  + http://idea.imsxm.com

2nd Class

* Naming convention
  + In folder/file name use Hyphen(-)
* HTML?
  + *Tag based markup language.* Make **Structure & content** in any web apps.
  + **HTML elements**: An element in HTML represents some kind of structure that consists of a start tag, content, and an end tag.
    - Ex: <p>This is the content of the paragraph element.</p>
    - 2 type of Element in HTML
      * **Block Level**; e.g. <p>, <div>
      * **IN line**;e.g: <span>, <a>, <img>
  + **HTML attributes:** 
    - Attribute provide additional information of a tag. (Like Body spray )
    - An attribute defines a property for an element.
    - <img src="foobar.gif" alt="A foo can be balanced on a bar by placing its fubar on the bar's foobar.">
  + **Tag:** 
    - Tag is a Syntax, used to shows any content in the browser.
    - define how the content looks like small or big
    - are used to mark up the start and end of an HTML element.
    - paragraph start tag: <p>, paragraph send tag: </p>
    - What is the syntax for tag writing ?
      * <Tag-name>Content</Tag-name> ,
    - ECO/ Google friendly <Tag-name />
    - Total 118+ Tag in HTML
  + Markup language ?
    - All Tag is called markup language. Such as all letter is called alphabet.
  + Sir **Tim Berners-Lee** created HTML-1 in 1991.
  + W3C(World Wide Web Consortium) define the tag validation/ version.
  + HTML -5 created 2008. W3C published in 2012. All process completed in 2014.
  + Browser is the server of HTML.
* CSS
  + Sir **Hakon wium Lie** created CSS-1 in 1996.
  + Class based style sheet Language
  + Basic structure/syntax : selector { Property: value }(curly-end)
  + ID - Not Duplicate, Class - Duplicate support
* <!DOCTYPE html>
  + HTML5 Declaration
  + This Syntax tell the browser that we use HTML5 Syntax
  + <html>
    - Mother tag of HTML
  + <table>
    - table expect the highest columns in every rows.
    - colspan is an attribute.
    - valign = top : to consider default padding of td element

3rd Class

Structure & Content of Any Web page are Created By HTML;

* In any article 3 things - Image, heading, content.
  + I have to use <h1> to <h6> tag for using heading
  + if i use <p> tag, that not ECO standard.
* **Keyword**(Used as keyword, site indexing): When we visit our site by any search engine, then SE send A technology in our site such as -
  + **Spider**: A **spider** is a program that visits Web sites and reads their pages and other information in order to create entries for a **search engine** index.
  + **crawler:** A **Web crawler**, sometimes called a **spider**, is an [Internet bot](https://en.wikipedia.org/wiki/Internet_bot) that systematically browses the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), typically for the purpose of [Web indexing](https://en.wikipedia.org/wiki/Web_indexing) (*web spidering*).
  + Spider visit in every page. Then take the content in some special tag such as -
    - <em>Emphasized text</em><br>
    - <strong>Strong text</strong><br>
    - <code>A piece of computer code</code><br>
    - <samp>Sample output from a computer program</samp><br>
    - <kbd>Keyboard input</kbd><br>
    - <var>Variable</var>
  + in their basket. Search engine used these **Important word** as a search hint and take in their server. Such as: Google take the keyword in google server. If no keyword set, SE indexing by default. When we search by thouse keyword, search engine match by their database and return our site.
* **Float:** Using float we can set multiple element side by side. 4 types of Alignment
  + Left-Right-Center-Justify
* Bold a Selected Article
  + <b></b>
    - Just design
  + <strong></strong> (Used as Keyword)
    - Keyword, important word, site indexing, SEO, spider, clural visit in every page and take keyword in their server
* Underline Selected Article; e.g:
  + <u></u>
  + <ins></ins> (Used as Keyword)
* Italic Selected Article
  + <i></i>
  + <em></em> (Used as Keyword)
* Article
  + Highlighting Selected Article; e.g: <mark></mark>
  + Writing Mathematical Article Using Superscript; e.g: <sup></sup>
  + Writing Chemistry Article Using Subscript; e.g: <sub></sub>
  + Writing Poetry Article Using Preformatted Text Tag; e.g: <pre></pre>
  + Writing Article Using anchor Formated Text Tag; e.g:
    - Developer used anchor tag in 3 ways :
      * Link in another site
      * Multiple Link in a same site
      * Link in same page upper or down
    - <a href="http://bitm.org.bd" ></a>
      * Basic structure
      * href = hyper reference
      * send bitm.org.bd request to server through http protocol
    - <a href="" target="\_blank"></a> (Open Linked Page in New Tab)
    - <a href="filename.html">Name</a> (Menu Button Creation)

* Project:
  + Project Folder = Your Name-SEIP ID
  + Portfolio Website:
    - 1. Home: (Article Writing) - a) Mission b) Vision c) Dream (10 Dreams)
    - 2. About - a) About Myself, b) My Familly, c) My Friends
    - 3. Portfolio - a) Complete CV,
    - 4. Gallery - a) My Choice of Photo
    - 5. Contact - a) My Contact Address

4th Class

* GIT
  + Go to the Help in github.com; e.g: https://help.github.com/
* Git command
  + git config --global user.name "Sanaulla"
  + git config --global user.email "sanaulla.ict@gmail.com"
  + git config --list
  + git clone https://github.com/SanaullaParvez/BITM-PHP.git
  + git status
* **DIV vs. Table**
  + div is for manage alignment and table is for manage content.
* **height = height + border + padding**, to escape this use -
  + **box-sizing: border-box**. Implement the value of width & height property.
* GRID System:
  + A **technique** of Layout that allows up to 12 columns across the page.
  + Make smooth of responsive web design.
* To define a common property in multiple class use -
  + **selector: [class\*=’col-’]{ }**

5th Class

element selector:feature{

property: value

}

* **Selector:** Selector used to indicate an specific element of HTML using class, id or tag name.
* **Pseudo class**:
  + A CSS pseudo-class is a keyword added to a selector that specifies a special state of the selected element(s)
  + A pseudo-class is used to define a special state of an element.
    - Ex: selector:hover, selector:after
* **Pseudo element**:
  + A CSS pseudo-element is used to style specified parts of an element.
    - Ex: selector:hover::**first-letter**
* **Combinators**:
  + A combinator is something that explains the relationship between the selectors.
    - Ex: div > p, div p, div + p, div ~ p,
* **Keyframes**: animation,
* Property
  + **display: block** : Any tag can absorb the full space of his parent.
  + **border-radius:** shape border corner.
    - Circle : height = weight = (>= [border-radius/2]/50%)
  + **transition**: 2s;
  + **transform**: skew(30deg);
  + **border**: ridge, outset, inset
* Global selector : **\*{ }**
* **wrapper/ container**: To minimum the content power in body. According to standard practice, in every element use minimum 3 layer without body. EX: wrapper > row > col-\*
* **NAV**: it’s easy for the search engine to understand the navigation tag.

6th Class

* **Media query**:a technique of css
  + apply css condition using the resolution of any device/ browser
  + screen
  + Ctrl + Shift + M => For mobile view in Mozilla
  + condition :
    - max-height: 500px
      * if (width <=500) then true else false;
    - min-height: 500px
      * if (width >=500) then true else false;
* **Bootstrap**
  + A complete structure of css
  + A skeleton of css
  + Framework of css
  + Framework of front-end
  + used for responsive web design
  + Developed by Mark Otto & Jacob Thornton in 2010, released in 2011, best project at github in 2012 is bootstrap
  + Other framework Ex: foundation, startup,
  + Alpha - not compile, Beta- Compiled
  + .css: Human readable; .min.css: not human readable, compressed, minified version
  + JS work for HTML & CSS, for that script tag
  + **Framework installation**
    - download, extract, load and link
  + Version: 2, 3(2014-2015), 4 ()
  + Bootstrap create **xl, lg, md, sm, col** classes by using grid system.
    - xl - extra large devise
    - lg - large device, such as: desktop
    - md - medium ,such as: laptop
    - sm - small, such as: tab
    - col - extra small, such as : mobile (In Bootstrap 3 : col-xs)

7th Class

* To develop any software
  + 1st create Structure
  + then add content
  + at last functionalities
* 3 click behavior used in form. Such as -
  + anchor -
    - to move to a new page or download a file, then use an anchor
  + input -
    - use inputs for submitting and resetting forms
  + button -
    - custom behavior
    - 3 types
      * reset
      * submit
      * button
* Input Element:
  + Why? To get input from user
  + Type? checkbox, radio
  + Attribute ? name - what user input - Naming Concept = first\_name (NOT f\_name)
  + **How many input element in html-5** ? 6( SIX )
    - input
    - textarea
    - select
    - data list - In Google -> Search -> Google Server( ) -> response -> suggestion
    - keygan
    - output
* Bootstrap class for Margin: Using margin we set a distance between div tag
  + mb-0
  + mx-0
  + my-0
* Bootstrap class for Typography/Text class
  + text-info
  + text-dark
  + text-secondary
* Bootstrap class for input element in a form
  + - form-group
      * form-control
* Bootstrap class for Button: 10 classes. such as -
  + btn-info
  + btn-danger
  + btn-dark ...
* Bootstrap class for Button-Outline: 8 classes. such as -
  + btn-outline-light
  + btn-outline-danger
  + btn-outline-warning ...
* Bootstrap class for Table
  + table
  + table-bordered
  + table-info
  + table-hover
  + table-responsive …
* Bootstrap class for Image
  + img-thumbnail
  + img-fluid
  + rounded
* Bootstrap Class in Margin & Padding
  + mr - margin right
  + mb - margin bottom
  + mx - x margin
  + pt - padding top
  + pr - padding right
  + pr-lg-0 - large devise
* Container -
  + class=”container” - Middle Width
  + class=”container-fluid - Full Width

8th Class

* PHP > apache web server > return browser
* User input data of 2 type:
  + Raw data: first name, email
  + File Data:
    - All Image. Ex: jpg, png -
      * type = ”image”
      * **accept="image/\*" - Browse only image**
      * **multiple - Select multiple images**
    - word file, pdf file
* **Blockquote**
  + **blockquote.blockquote**
  + **footer.blockquote-footer**
* Form
  + Vertical From
    - div.**form-group**
      * label
      * **input.form-control**
  + Horizontal From
    - div.**form-group.row**
      * label.**col-form-label col-sm-3**
      * div.**col-sm-9**
        + **input.form-control**
  + Inline From
* **Many of our components require the use of JavaScript to function. Specifically, they require jQuery, Popper.js, and our own JavaScript plugins.**
  + **1. jQuery - http://jquery.com/download/**
  + **2. Pooper - getbootstrap.com > Download > pooper.min.js**
  + **3. Bootstrap.min.js**
    - **Modal - Modal.js** 
      * **data-toggle="modal"**
    - **Dropdown - Dropdown.js** 
      * **data-toggle="dropdown"**
    - **Carousel - Slider Images** 
      * **data-ride="carousel"**
      * **data-interval="1800”**
      * **data-slide="prev"**
      * **data-slide-to="0"**
      * **data-dismiss="modal"**
    - **Collapse - Collapse.js** 
      * **data-toggle="collapse"**
    - **Data target - Open and close**
      * **data-target="#bitm"**
* Navigation
  + **nav.navbar**
  + **navbar-expand-sm/ navbar-expand-lg - If menu item is minimum/ maximum**
  + **bg-dark navbar-dark - For Background Color**
  + **sticky-top - If navbar is fixed when scroll**
    - **div.container - Need when multiple ul.navbar-nav**
      * **LOGO - a.navbar-brand**
      * **button.navbar-toggler data-toggle="collapse" data-target="#bitm"**
        + **span.navbar-toggler-icon**
      * **div.collapse.navbar-collapse#bitm**
        + **ul.navbar-nav**
        + **ul.mr-auto - When Right Button in the Right SIde**

li.nav-item

a.nav-link

**Dropdown -** li.nav-item dropdown

**a.nav-link.dropdown-toggle data-toggle="dropdown"**

**ul.dropdown-menu**

**li.nav-item**

**a.nav-link**

* **Jumbotron**
  + **div.jumbotron**
    - **div.container**
      * **h3**
      * **p**

9th Class

* **Carousel**
  + **div.carousel - initialize(as wrapper)**
  + **div.slide - Slide from left to right**
  + **data-ride="carousel" - Basic for slide**
  + **data-interval="1800” - Slide after 1 second**
  + **#mainSlide - To call as data-target=”#....” or href=”#....”**
  + **1. inner - Important**
    - **div.inner data-ride="carousel" - Image container**
      * **div.carousel-item.active - Wrap an Image**
        + **image**
        + **div.carousel-caption - T0 Write Content**

**<h3/> , <p></p>**

* + **2. control**
    - **a.carousel-control-prev**
    - **href="#mainSlide"**
    - **data-slide="prev" -** 
      * **span.carousel-control-prev-icon**
    - **data-slide="next" -** 
      * **span.carousel-control-next-icon**
  + **3. indicators** 
    - **ol.carousel-indicators**
      * **li data-target="#mainSlide" data-slide-to="0"**
* Modal
  + div.**modal - Wrapper**
  + **.fade - To Fade In & Fade out**
    - div.**modal-dialog.modal-lg - Modal Size , Ex: Lg, md, sm**
      * div.**modal-content - Wrapper of 3 Parts**
        + 1. div.**modal-header**

**H3**

**button class="close" data-dismiss="modal"**

**&Cross;**

* + - * + 2. div.**modal-body**
        + 3. div.**modal-footer**
* Card - A card is a flexible and extensible content container. Represent Article in Browser.
  + Real Example: http://www.prothom-alo.com/
  + div.card
    - 1. div.**card-header /** img.card-img-top
    - 2. div.**card-body**
      * h4.card-title
      * p.card-text
      * a.btn
    - 3. div.**card-footer**
* Section
  + Background
    - **background-image**: **url**(**"../images/3.jpg"**); - directory
    - **background-repeat**: **no-repeat**; - not duplicate
    - **background-size**: 100% 100%; - Image in Full space & responsive
    - **background-size**: cover; - Image in Full space & but not responsive
    - **background-attachment**: **fixed**; - Image fixed But content scroll
* Accordion

10th Class

* JS
  + can get error in any code - Programming language
  + No machine language - not programming language
  + Object oriented scripting language -
  + 1994 - sir onded ud
* JS: Communication Between Client and Web Content(HTML/ TAG).
  + Control HTML Element **Behaviour**
  + Partial Programming Language
* Every HTML Element has -
  + Default Characteristics
  + Content
  + Attribute

In Raw JavaScript - 6 Syntax

* **1. Syntax**
  + Camel Case - getElementById
* **2. Variable** - Memory location/ address -
  + Naming Convention: 3 major rules
    - start var sign
    - A\_Z, a-z, 0-9, \_, $
    - no number in first
    - Ex: var myName = “Sanaulla”
  + 3 Standard Rules
    - Small letter
    - meaningful
    - readable
  + loosely coupled type - NO need to define type like Java
    - typeOf(firstName)

* **3. Operator**
  + 1. Arithmetic
    - Binary (+ - \* / %)
    - Unary (++, --, (-))
  + Assignment (+= -= \*= /= %=)
  + Relational**/** Conditional (> >= != == === !===)
  + Logical [ AND((&&),OR(||), NOT( ! ) ]

11th Class

* **4. Statement** - Any Line of Programming Language(Close By Semi Clone(**;**))
  + 3 type of statement
    - 1. Single Line Statement
      * Execute By Run in Browser
    - 2. Conditional Statement
      * If condition true then execute
      * if, if...else, if...else….if - depending on different condition
      * switch - depending on any value
    - 3. Repeated Statement
      * execute multiple time
      * 4 - for, while, do...while,
        + for...in - By index value defined in array

var x = new Array()

var y = Array();

var z = [];

* + 5 things in a statement
    - I. Value
      * Variable value. Ex: x,y,z.
      * Fixed Value Ex: PI, 10.
    - II. syntex
      * Reserve Word. Ex: var, for, while ...
    - III. Operator
      * +, -, \*, /
    - VI. Expression
      * value+operator>=expression
    - V. Comments

12th Class

* **5. Array**
  + **var paragraphs = document.getElementsByTagName('p');**
  + **for(var eachParagraph in paragraphs){**
  + **document.write(paragraphs[eachParagraph].innerHTML);**
  + **document.write('<br/>');**
  + **}**
  + **for(x=0;x<=paragraphs.length;x++){**
  + **document.write(paragraphs[x].innerHTML+'<br/>')**
  + **}**
* **6. Function - a Syntex** (So many function in your mobile)
  + Block of Code
  + Particular Jobs - Have Any activity (tar ekta kaj ace)
  + Re-use
  + Can not work himself
  + have to call
  + maximum can return
* Function -
  + 2 part
    - Calling
      * summation(argument);
      * when function call - argument
    - Definition
      * function summation(){ }
        + parenthesis - ()
        + carly start, carly close - {}
        + syntax - function
        + name - demo
        + scope ()

In middle of scope - parameter

* + - * + body - {middle of this}
      * Dynamic function - when parameter define
  + 2 type
    - user define -
    - built in -
      * alert()
      * sort
      * join
      * write
      * confirm
      * date
      * time
      * math
* Event
  + **Event** - choto choto gotona
    - Who Execute - User/ Visitor/ Client -
    - Where Execute - In HTML Element
  + **Event attribute** - the name of event
    - the definition of an event
    - scroll, type, click,
    - HTML -er **o** dia joto attribute ace
      * onclick, oncancel, onblur, onscroll ….. - total 48 attribute
  + **Event attachment/Event Listener in HTML Element** -
    - NEED TO ATTACH ON THE HTML ELEMENT
    - Ex: myHeader.onclick
  + 3 way to Execute
    - 1. In HTML Tag - onclick
      * onclick="**this**.style.color = 'blue';"
      * [According to standard practice it not use]
    - 2. Set an ID
      * myHeader.onclick = **function** () {
      * **this**.style.color = 'red';
      * **this**.style.backgroundColor = 'blue';
      * }
    - Anonymou function - If no name in any function;
      * Call itself as recursive function

13th Class

* JS
  + Using Event, Generate communication between Client and Web Content.
* DOM
  + Document Object Model: By following A set of Model/rules & Using Object, We Create Document.
  + Concept/ Perception/ Standard
  + A Single page is a Document
  + Individual Existence Pieces - Object
  + Set of Standard/ Rules - Model
* Make a DOM Structure of any DOcument/ index.html page ??? Tag is Node.
  + HTML
    - Meta
    - Title
  + Body
    - h1
    - p
    - ul
* 3 Works in DOM
  + DOM Element Access
  + DOM Element Content Access
  + DOM Element Create
* BOM
  + Browser Object Model

14th Class

* jQuery
  + js Library - http://api.jquery.com/
    - Library - Complete Class
      * Class - Syntax of OOP
        + number of property - single variable/ information
        + & method - function/ activity
        + OOP - Programming er Model

In Define - Variable

Class- Property

OOP - Attribute

In Define - function

Class - method

OOP - behaviour

* + Facility
    - Cross-Browser-Issue: If we use a Property or Method that define in Chrome but not in Opera, Explorer. SO use jQuery that define all Property or Method in a jQuery file.
    - Minimum code writing
    - Huge Built-in Plugin, Customization
  + $(Selector).Action()
    - $ - JQuery Object
    - Selector - HTML Element Selector - Syntax
    - . - jQuery Object sign
    - Action() - jQuery method
      * Normal method - Ex: css(),
      * Event Handler Method - Ex: click()
  + 2 ways To Store Data:
    - JSON- js Object Notation
    - XML - Extensible Markup Language

<!--XML format employee INFO-->

<employee>

<firstname>Sanaulla</firstname>

<lastname>Parvez</lastname>

<city>Dhaka</city>

<salary>45000</salary>

</employee>

<!--json FORMAT EMPLOYEE INFO-->

{

'first-name': 'Sanaulla',

'last-name': 'Parvez'

}

* 3 ways to get DOM Element Content
  + text() , html()
    - document.getElementById('#first\_name').innerHTML() - JS
    - $('#first\_name').text(); - jQuery
  + val()
    - document.getElementById('#first\_name').value - JS
    - $('#first\_name').val(); - jQuery
  + attr(‘src’)
    - document.getElementById('#first\_name').getAttribute('src');- JS
    - $('#main-image').attr('src',newImagePath);- jQuery
* Form: Transfer Data From Client Machine to Server
  + How Carry ? FORM Method
    - GET : URL - Fast, Less Security, Total 255 Characters
    - POST : Hidden - Slow, Unlimited Data Transfer, Secured
  + Which Path ? - Action
  + Event Handler Method Attach korlam …
    - $(‘’).keyup();

15th Class

* Web App Development - 3 Parts
  + Front End
  + Server
  + DB
* Rasmus Lerdorf -
  + History - https://en.wikipedia.org/wiki/PHP
  + created the PHP - 1994;
  + Press conference - 1995
  + Personal Home Page
  + 1998 - PHP 3
  + 2000 - PHP 4
  + 2004 - PHP 5
    - Interface
    - Abstract
    - Polymorphism
    - 2014 - PHP 5.6
  + 2015’Dec - PHP 7
    - Same as JAVA/ C#
    - 7.0, 7.1,
    - 7.2 - 2017
* Taylor Otwell - Created the Laravel
* AMC - Annual Maintenance Cost
* PHP ?
  + Official Website - <http://php.net/>
  + widely-used
  + open source
  + general-purpose scripting language that is especially
  + suited for web development and can be
  + embedded into HTML.
  + compiler of php/ server of php
    - Apachee
    - IIS - Windows
    - Geus -
    - Aol Server

16th Class

* Desktop Base Software - To avoid Terminal base work in PHP/ GUI
  + Xampp (More facility)/ Wamp - My PC as Local server - Download Path : https://www.apachefriends.org/download.html
    - Apachee - PHP Server
    - Mysql - Database
    - Filezilla - Upload
    - Merquery - email
    - Tomcat - C#
  + Computer Memory
    - Harddisk - volatile
    - RAM - non volatile - When run any program - use
* PHP
  + **syntax**
    - \*.php <?php ?> -
  + **variable**
  + **operator**
  + **statement**
  + **array**
  + **function**
  + **class**

purchase / buy

sale

sudo apt-get install php5.6-mbstring

sudo apt-get install php5.6-xml

sudo apt-get install php5.6-sqlite

cd /var/www/html/laravel5/ && tail -f storage/logs/laravel.log

cd /var/www/laravel/ && tail -f storage/logs/laravel.log

sudo composer create-project laravel/laravel /var/www/laravel 4.2

composer create-project laravel/laravel blog "5.1.\*"

php artisan key:generate

sudo composer create-project --prefer-dist laravel/laravel sadi "5.1.\*" --no-scripts

sudo composer update --prefer-dist

sudo chmod -R 775 /var/www/laravel/app/storage

sudo chown -R fedena:www-data /var/www/laravel/app/storage

php artisan make:controller WelcomeController

php artisan make:model Car --migration

php artisan migrate:make create\_users\_table --table="users" --create

php artisan migrate:make add\_email\_to\_users\_table --table="users"

php artisan migrate

php artisan migrate:rollback

php artisan cache:clear

php artisan serve

php artisan routes

php artisan down // ON DEVELOPMENT MODE

composer require gladcodes/keygen

composer require patricktalmadge/bootstrapper

<?php

public function getRules() {

return $this->rules

}

$duck = new Duck;

$duck->getRules();

//Injest User class through controller constructor

{{dd($users->toArray())}}

{{link\_to("users/{$user->username}", $user->username)}}

<a href="{{ URL::to('logout') }}">Logout</a>

@if(isset($name))

{{$age}} <br/>

@else

{{$name}} <br/>

@endif

{{ $age }} <br/>

{{ $location }} <br/>

Routes

---------------------------------------------------------------------

Route::get('users/{username}',function($username)

{

$user = User::whereUsername($username)->first();

return View::make('users.show',['user' => $user]);

});

Route::get('users.index',function()

{

$users = User::all();

return View::make('demo1.user')->with('users',$users);

return View::make('demo1/user')->withUsers($users);

return View::make('users/users',['users' => $users]);

return Redirect::back()->withInput();

return Redirect::route('sessions.create');

return Redirect::to('/admin');

});

Route::get('/', function()

{

$user = DB::table('users')->where("username", "!=", "sanaulla")->get();

$user = User::where("username", "!=", "sanaulla")->get();

return $user;

$user = new User;

$user->username = 'Asif';

$user->password = Hash::make('Asif');

$user->save();

User::create([

'username' => 'Forhad',

'password' => Hash::make('Asif')

]);

$user = User::find(1);

$user->username = 'Amin';

$user->save();

$user = User::find(1);

$user->delete();

$users = User::orderBy('username', 'asc')->take(2)->get();

return $users;

return $user->email;

});

?>