**Department of Information Technology**

**List of Assignments**

**Subject**: Laboratory Practice-II(WAD) **Class: TE (IT)**

**Teaching Scheme: Examination Scheme:**

**Practical: 2 Hours/Week Practical: 25 Marks**

**Term Work: 25 Marks**

|  |  |
| --- | --- |
| **Sr. No.** | **Title of Lab Assignment** |
| 1 | **a**. Create a responsive web page which shows the ecommerce/college/exam admin dashboard with sidebar and statistics in cards using HTML, CSS and Bootstrap.  b. Write a JavaScript Program to get the user registration data and push to array/local storage with AJAX POST method and data list in new page. |
| 2 | a. Create version control account on GitHub and using Git commands to create repository and push your code to GitHub.  b. Create Docker Container Environment (NVIDEIA Docker or any other).  c. Create an Angular application which will do following actions: Register User, Login User, Show User Data on Profile Component |
| 3 | a. Create a Node.JS Application which serves a static website.  b. Create four API using Node.JS, ExpressJS and MongoDB for CURD Operations on assignment 2.C. |
| 4 | a. Create a simple Mobile Website using jQuery Mobile.  b. Deploy/Host Your web application on AWS VPC or AWS Elastic Beanstalk. |
| 5 | Mini Project  Develop a web application using full stack development technologies in any of the following domains:  1. Social Media  2. ecommerce  3. Restaurant  4. Medical  5. Finance  6. Education  7. Any other |



**COURSE OBJECTIVES :**

**1.** To understand basic concepts of web programming and scripting languages.

**2.** To learn Version Control Environment.

**3.** To learn front end technologies and back end technologies.

**4.** To understand mobile web development.

**5.** To comprehend web application deployment.

**COURSE OUTCOMES:**

On completion of the course, students will be able to–

**CO1:** Develop Static and Dynamic responsive website using technologies HTML, CSS, Bootstrapand AJAX.

**CO2:** Create Version Control Environment.

**CO3:** Develop an application using front end and backend technologies.

**CO4:** Develop mobile website using JQuery Mobile.

**CO5:** Deploy web application on cloud using AWS.

**Assignment 1a**

Title- Create a responsive web page which shows the ecommerce/college/exam admin dashboard with sidebar and statistics in cards using HTML, CSS and Bootstrap.

Theory:

* 1. HTML
* HTML stands for Hyper Text Markup Language
* HTML is the standard markup language for creating Web pages
* HTML describes the structure of a Web page
* HTML consists of a series of elements
* HTML elements tell the browser how to display the content
* HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc

A simple HTML Document

|  |
| --- |
| <!DOCTYPE html> <html> <head> <title>Page Title</title> </head> <body>  <h1>My First Heading</h1> <p>My first paragraph.</p>  </body> </html> |

* The <!DOCTYPE html> declaration defines that this document is an HTML5 document
* The <html> element is the root element of an HTML page
* The <head> element contains meta information about the HTML page
* The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
* The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
* The <h1> element defines a large heading
* The <p> element defines a paragraph
  1. CSS
* CSS stands for Cascading Style Sheets
* CSS describes how HTML elements are to be displayed on screen, paper, or in other media
* CSS saves a lot of work. It can control the layout of multiple web pages all at once
* External stylesheets are stored in CSS files
* CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

Simple CSS example

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <style>  body {  background-color: lightblue;  }  h1 {  color: white;  text-align: center;  }  p {  font-family: verdana;  font-size: 20px;  }  </style>  </head>  <body>  <h1>My First CSS Example</h1>  <p>This is a paragraph.</p>  </body>  </html> |

* + 1. Bootstrap

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones).

Why Bootstrap?

* Faster and Easier Web Development.
* It creates Platform-independent web pages.
* It creates Responsive Web-pages.
* It is designed to be responsive to mobile devices too.
* It is Free! Available on [www.getbootstrap.com](http://www.getbootstrap.com)
* Simple Example

|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>   <title>Bootstrap Example</title>   <meta charset="utf-8">   <meta name="viewport" content="width=device-width, initial-scale=1">   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script> </head> <body>  <div class="container">   <h1>My First Bootstrap Page</h1>   <p>This is some text.</p> </div>  </body> </html> |

Bootstrap CDN

|  |
| --- |
| <!-- Latest compiled and minified CSS --> <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">  <!-- jQuery library --> <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>  <!-- Latest compiled JavaScript --> <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script> |

Conclusion- Thus we have created a responsive web page which shows the admin dashboard with sidebar and statistics in cards using HTML, CSS and Bootstrap.

**Assignment 1b**

Title- Write a JavaScript Program to get the user registration data and push to array/local storage with AJAX POST method and data list in new page.

Theory:

We can save arrays and objects to browser storage using JavaScript as an alternative to using cookies with couple of ways.

The Web Storage API allows us to store items in a more intuitive way using two mechanisms: localStorage and sessionStorage.

For both of these mechanisms:

• Items are converted to a JSON string using JSON.stringify().

• Items are stored in key-value pairs.

• When items are retrieved from storage, the JSON string is parsed using JSON.parse().

• When items have been removed from storage, the storage can be cleared.

**The localStorage API**

The localStorage API maintains browser storage even when the browser is closed and reopened.

The code example below shows how the localStorage API is used to store arrays:

const myArray = ['a', 'b', 'c', 'd'];

// convert array to JSON string using JSON.stringify()

const jsonArray = JSON.stringify(myArray);

// save to localStorage using "array" as the key and jsonArray as the value

localStorage.setItem('array', jsonArray);

// get the JSON string from localStorage

const str = localStorage.getItem('array');

// convert JSON string to relevant object

const parsedArray = JSON.parse(str);

console.log(parsedArray);

To save objects, replace myArray with something like myObject. We’ll use the sessionStorage API to demonstrate saving objects for brevity.

**The sessionStorage API**

The sessionStorage API stores items for as long as the browser window is open, including page reloads and restores. The data is lost once the browser window is closed.

Here’s a code example to show how to use the sessionStorage API to store objects:

const Person = {

name: 'John Smith',

age: 18,

};

// convert object to JSON string using JSON.stringify()

const jsonObject = JSON.stringify(Person);

// save to localStorage

sessionStorage.setItem('person', jsonObject);

// get the JSON string from sessionStorage

const str = sessionStorage.getItem('person');

// convert JSON string to valid object

const parsedObject = JSON.parse(str);

console.log(parsedObject);

To remove items from local and session storage, use the removeItem() method, giving it the key, as follows:

localStorage.removeItem('array');

sessionStorage.removeItem('person');

The local and session storage can also be cleared using clear():

localStorage.clear();

sessionStorage.clear();

**AJAX**

AJAX stands for Asynchronous JavaScript and XML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script.

Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.

Conventional web applications transmit information to and from the sever using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.

With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the

results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server. XML is commonly used as the format for receiving server data, although any format, including plain text, can be used. AJAX is a web browser technology independent

of web server software.

AJAX is based on the following open standards −

* Browser-based presentation using HTML and Cascading Style Sheets (CSS).
* Data is stored in XML format and fetched from the server.
* Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
* JavaScript to make everything happen

AJAX cannot work independently. It is used in combination with other technologies to create interactive webpages.

• JavaScript

– Loosely typed scripting language.

– JavaScript function is called when an event occurs in a page.

– Glue for the whole AJAX operation.

• DOM

– API for accessing and manipulating structured documents.

– Represents the structure of XML and HTML documents.

• CSS

– Allows for a clear separation of the presentation style from the content

and may be changed programmatically by JavaScript

• XMLHttpRequest

– JavaScript object that performs asynchronous interaction with the server.



Fig 1. How AJAX works

**AJAX – Events : onreadystatechange Event Properties**

|  |  |
| --- | --- |
| Property | Description |
| onReadyStateChange | It is called whenever readystate attribute changes. It must not be used with synchronous requests. |
| readyState | represents the state of the request. It ranges from 0 to 4.   * 0: request not initialized (open() is not called.) * 1: server connection established (open is called but send() is not called.) * 2: request received (send() is called, and headers and status are available.) * 3: processing request (Downloading data; responseText holds the data.) * 4: request finished and response is ready (The operation is completed fully.) |
| Status | 200: "OK" 403: "Forbidden" 404: "Page not found" |

XMLHttpRequest object properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| readyState | An integer from 0. . .4. (0 means the call is uninitialized, 4 means  that the call is complete.) |
| onreadystatechange | Determines the function called when the objects readyState changes. |
| responseText | Data returned from the server as a text string (read-only). |
| responseXML | Data returned from the server as an XML document object (read-only). |
| status | HTTP status code returned by the server |
| statusText | HTTP status phrase returned by the server |

XMLHttpRequest object methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| open('method', 'URL', asyn) | Specifies the HTTP method to be used (GET or POST as a string, the target URL, and whether or not the request should be handled asynchronously (asyn should be true or false, if omitted, true is assumed). |
| send(content) | Sends the data for a POST request and starts the request, if GET is used you hould call send(null). |
| setRequestHeader('x','y') | Sets a parameter and value pair x=y and assigns it to the header to be sent with the request. |
| getAllResponseHeaders() | Returns all headers as a string. |
| getResponseHeader(x) | Returns header x as a string. |
| abort() | Stops the current operation. |

Sample code :

ajaxcommunication.html

|  |
| --- |
| <html>  <body>  <div id="xyz">  Hello Friends <br>  Welcome to Pune!!!!!<br>  <button type="button" onclick="load()">  Submit  </button>  </div>  <script>  function load(){  var req=new XMLHttpRequest()  req.onreadystatechange=function() {  if(this.readyState == 4 && this.status == 200){  document.getElementById("xyz").innerHTML=this.responseText  }  }  req.open('GET','data.txt',true)  req.send()  }  </script>  </body>  </html> |

Data.txt

|  |
| --- |
| I am enjoying learnimg JavaScript!!!!!! |

Conclusion: Thus, JavaScript Program is implemented to get the user registration data and we pushed it to local storage with AJAX POST method and listed the data in new page.

**Assignment 2a**

Title- Create version control account on GitHub and using Git commands to create repository and push your code to GitHub.

Theory:

1. What is Git?

Git is a popular version control system. It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

* Tracking code changes
* Tracking who made changes
* Coding collaboration

2. What does Git do?

* Manage projects with Repositories
* Clone a project to work on a local copy
* Control and track changes with Staging and Committing
* Branch and Merge to allow for work on different parts and versions of a project
* Pull the latest version of the project to a local copy
* Push local updates to the main project

3. Working with Git

* Initialize Git on a folder, making it a Repository
* Git now creates a hidden folder to keep track of changes in that folder
* When a file is changed, added or deleted, it is considered modified
* You select the modified files you want to Stage
* The Staged files are Committed, which prompts Git to store a permanent snapshot of the files
* Git allows you to see the full history of every commit.
* You can revert back to any previous commit.
* Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!

4. Why Git?

* Over 70% of developers use Git!
* Developers can work together from anywhere in the world.
* Developers can see the full history of the project.
* Developers can revert to earlier versions of a project.

5. What is GitHub

* Git is not the same as GitHub.
* GitHub makes tools that use Git.
* GitHub is the largest host of source code in the world, and has been owned by Microsoft since 2018.

6. Steps to Push and PULL version control repository to GitHub

|  |  |  |
| --- | --- | --- |
| Step No | Command | Description |
| 1 | Git Installation | Download Git from the website: [https://www.git-scm.com/](https://git-scm.com/) |
| 2 | Command line  >git –version | If Git is installed, it should show something like git version X.Y |
| 3 | git config --global user.name "w3schools-test"  git config --global user.email "test@w3schools.com" | Configure Git  Change the user name and e-mail address to your own |
| 4 | mkdir myproject  cd myproject | Creating Git Folder |
| 5 | git init | Initialize GitInitialized empty Git repository in /Users/user/myproject/.git/ |
| 6 | git status | To check the status |
| 7 | git add index.html | Add file to staging environment |
| 8 | git add --all | add all files in the current directory to the Staging Environment: |
| 9 | git commit -m "First release of Hello World!" | The committ command performs a commit, and the -m "message" adds a message. |
| 10 | git commit -a -m "Updated index.html with a new line" | Skips staging environment |
| 11 | git log | To view the history of commits for a repository, you can use the log command |
| 12 | git *command* -help | See all the available options for the specific command |
| 13 | git help --all | See all possible commands |
| 14 | git commit -help | See help for specific command |
| 15 | git branch hello-world-images | a branch is a new/separate version of the main repository. This command creates a new branch hello-world-images |
| 16 | git checkout hello-world-images | checkout is the command used to check out/ move to a branch |
| 17 | git checkout master | Used to switch between branches |
| 18 | https://github.com/ | Create a new account on github |
| 19 |  | Create a Repository on GitHub |
| 20 | git remote add origin https://github.com/w3schools-test/hello-world.git | Push Local Repository to GitHub |
| 21 | git push --set-upstream origin master | push master branch to the origin url, |
| 22 |  | go back into GitHub and see that the repository has been updated: |
| 23 | git fetch origin | fetch gets all the change history of a tracked branch/repo |
| 24 | git merge origin/master | merge combines the current branch, with a specified branch. |
| 25 | git pull origin | pull is a combination of fetch and mergeIt is used to pull all changes from a remote repository into the branch you are working on. |

**Conclusion-Thus we have c**reated a version control account on GitHub and used Git commands to create a repository and pushed our code to GitHub.

**Assignment 2c**

Title- Create an Angular application which will do following actions: Register User, Login User, Show User Data on Profile Component

Theory-

Useful link - https://www.tutorialsteacher.com/angular/install-angular

1. Angular requires a current, active LTS(long term support) or maintenance LTS version of Node.js and NPM.

install node.js https://nodejs.org/

It will automatically install NPM - node package manager

1. Install Angular CLI

npm install -g @angular/cli@latest

To Create Angular 2 Application Angular CLI is required

1. To create new project

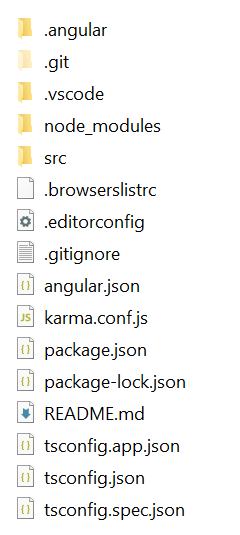
through CLI go to folder of the new project

Give command as -

ng new project-name

press ENTER

The project will be created as directory structure below –



Open folder src/app

Modify app.module.ts for form application

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import {FormsModule} from '@angular/forms'

@NgModule({

  declarations: [

    AppComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

Open app.component.html

Write html code for form (representative code is mentioned here, modify for multiple inputs)

<h1>Simple Form</h1>

<form #simpleForm = "ngForm" (ngSubmit) = "getValues(simpleForm.value)">

  <input type ="text" ngModel name = "user" placeholder = "Enter Name">

  <br> <br>

  <input type ="text" ngModel name = "age" placeholder = "Enter age">

  <br> <br>

  <input type ="text" ngModel name = "city" placeholder = "Enter city">

  <br> <br>

  <button>Get user value</button>

</form>

Make changes in app.component.ts

import { Component } from '@angular/core';

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'AngProj1';

  getValues(val:any)

  {

    console.log(val);

  }

}

Here getValue() function which is called in form file is defined.

You can check inputted values through form in console.

build application

1. Use Angular CLI command ng serve -o to build an application.

The -o indicates to open it automatically in the default browser.

2. Use NPM command ‘npm start’ to build an application

http://localhost:4200 to see the application home page.

3.Open the terminal in VS Code from menu Terminal -> New Terminal,

and type ng serve -o command and press enter,

You can send the form contents from console to other page.

On the basis of above implementation, you can design login user, show user data.

**Assignment 3**

a. Create a Node.JS Application which serves a static website.

b. Create four API using Node.JS, ExpressJS and MongoDB for CURD Operations on assignment 2.C.

Theory:

**Part a:**

**Create a Node.JS Application which serves a static website**.

Installation : Node.js (site – Node.js), Express.js(installed through cmd)

**Theory :**

**Node.js overview**

In basic terms, Node is an open source cross-platform library for server-side programming that permits clients to develop web applications rapidly. With Node, we can execute JavaScript applications or network applications. Its basic modules are engraved in JavaScript.

It is generally utilized for server applications in real-time. Node.js permits JavaScript to execute locally on a machine or a server.

Node.js gives numerous systems to utilize. One of such structures is Express.js. It is more valuable and mainstream than the different structures of Node.js.

**Features of Node.js**

* **Versatility:** Node is incredibly adaptable as the server reacts in a non-blocking way.
* **Zero Buffering:** Applications yield the measurements in enormous pieces. This gives the advantage of ‘No buffering’ to developers.
* **Network:** Node.js upholds an open-source community. This is the main explanation that numerous glorious modules have been added to Node.js applications over time.
* **Occasion driven Input and output:** APIs of Node.js are non-blocking, meaning that the server won’t wait for the arrival of information from an API. Rather, it will move to another API.

**Advantages of Node.js**

* **Easy to learn:** Node is quite simple for developers to utilize and learn. Learning Node.js is less difficult than React.
* **Better Performance:** Node.js takes the code of JavaScript via Google’s V8 JavaScript engine. The main advantage of this process is that it complies with the JavaScript code directly into the machine code
* **Freedom:** Node.js offers a lot of freedom when it comes to development. There are generally less constraints with Node.
* **Extended support for tools:** Another advantage of Node.js is that developers have more community support.
* **Extensible:** Node.js is known to be quite extensible. You can utilize JSON to give the degree to trade of information between the web server and the client.
* **Scalability:** Node.js makes it simple to scale applications in horizontal as well as vertical directions. The applications can be scaled even by the option of extra hubs to the current framework.

**Limitations of Node.js**

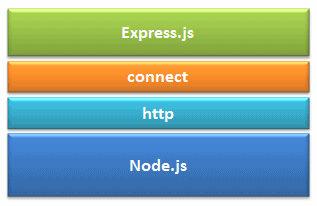
* **Programming interface isn’t steady:** The Application Programming Interface (API) of Node can be challenging to work with. It changes regularly and doesn’t remain stable.
* **No strong library support system:** JavaScript does not hold a strong library system. This limits the developers to implement even common programming tasks using Node.js.
* **Programming model is not synchronous:** Many developers find this programming model tougher in comparison to linear blocking I/O programming. In asynchronous programming, the codes become clumsier, and developers have to depend on the nes

# **Express.js**

"Express is a fast, unopinionated minimalist web framework for Node.js" - official web site: [Expressjs.com](https://expressjs.com/)

Express.js is a web application framework for Node.js. It provides various features that make web application development fast and easy which otherwise takes more time using only Node.js.

Express.js is based on the Node.js middleware module called ***connect*** which in turn uses **http** module. So, any middleware which is based on connect will also work with Express.js.

[](https://www.tutorialsteacher.com/Content/images/nodejs/expressjs.png)

Advantages of Express.js

1. Makes Node.js web application development fast and easy.
2. Easy to configure and customize.
3. Allows you to define routes of your application based on HTTP methods and URLs.
4. Includes various middleware modules which you can use to perform additional tasks on request and response.
5. Easy to integrate with different template engines like Jade, Vash, EJS etc.
6. Allows you to define an error handling middleware.
7. Easy to serve static files and resources of your application.
8. Allows you to create REST API server.
9. Easy to connect with databases such as MongoDB, Redis, MySQL

**Steps :**

1. Install Node.js
2. Setting up express.js
3. Structuring files
4. Creating your express server
5. Servicing your static files
6. Building your web page
7. Running your project

Open Node.js command terminal & run the following in your terminal-

Setting up express.js

1. Create a new directory for your project  - mkdir your-project-name
2. Change into your new directory - cd your-project-name
3. Initialize a new Node project with defaults. This will set a package.json file to access your dependencies: npm init -y
4. Create your entry file, index.js. This is where you will store your Express server: if you are working on Linux, you can run : touch index.js. if you are working on windows, you can edit in VSCode
5. Install Express as a dependency : npm install express –save
6. Edit package.json. Within your package.json, update your start script to include node and your index.js file.

Let express-static-file-tutorial is your project name

**Package.json**

{

"name": "express-static-file-tutorial",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"start": "node index.js" // change start value as node index.js

}, // This will allow you to use the npm start command in your terminal to launch

"keywords": [], your express server

"author": "Paul Halliday",

"license": "MIT"

}

## Structuring Your Files

To store your files on the client-side, create a public directory and include an index.html file

express-static-file-tutorial

|- index.js

|- public

|- index.html

## Creating Your Express Server

## Edit index.js file

Index.js

**const express = require('express');**

**const app = express();**

**const PORT = 3000;**

**app.use(express.static('public')); // represents application is serving static webpage in public directory**

**app.get('/', (req, res) => {**

**res.send('Hello World!');**

**});**

**app.listen(PORT, () => console.log(`Server listening on port: ${PORT}`));**

First of all, import the Express.js module.

In the above example, we imported Express.js module using require() function. The express module returns a function. This function returns an object which can be used to configure Express application (app in the above example).

The app object includes methods for routing HTTP requests, configuring middleware, rendering HTML views and registering a template engine.

The app.listen() function creates the Node.js web server at the specified host and port. It is identical to Node's http.Server.listen() method. Instead of Get(), post(), put() and delete() methods can be used.

## Building Your Web Page – client side

Navigate to your index.html file in the public directory. Populate the file with body and image elements:

[label index.html]

**<html>**

**<head>**

**<title>Hello World!</title>**

**</head>**

**<body>**

**<h1>Hello, World!</h1>**

**<img src="shark.png" alt="shark"> //download & store image in public directory**

**</body>**

**</html>**

(Instead of building Hello world application, building applications like student’s Registration form/main page of website is recommended)

## Running Your Project

In your terminal, launch your Express project

npm start

It will display

Server listening on port : 3000

Open your web browser, and navigate to <http://localhost:3000>.

## Conclusion

Assignment 4:

a. Create a simple Mobile Website using jQuery Mobile.

b. Deploy/Host Your web application on AWS VPC or AWS Elastic Beanstalk. Mini Project

Develop a web application using full stack development technologies in any of the following domains:

1. Social Media

2. ecommerce

3. Restaurant

4. Medical

5. Finance

6. Education

7. Any other

Theory:

**Part a:**

**jQuery Mobile**

JQuery Mobile is a user interface framework, built on jQuery Core and used for developing responsive websites or applications that are accessible on mobile, tablet, and desktop devices. It uses features of both jQuery and jQueryUI to provide API features for mobile web applications. This tutorial will teach you the basics of jQuery Mobile framework. We will also discuss some detailed concepts related to jQuery Mobile.

Why Use jQuery Mobile?

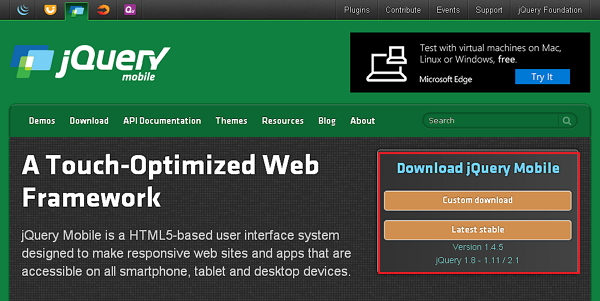
* It creates web applications that it will work the same way on the mobile, tablet, and desktop devices.
* It is compatible with other frameworks such as PhoneGap, Whitelight, etc.
* It provides a set of touch-friendly form inputs and UI widgets.

Features of jQuery Mobile

* It is built on jQuery Core and "write less, do more" UI framework.
* It is an open source framework, and cross-platform as well as cross-browser compatible.
* It is written in JavaScript and uses features of both jQuery and jQuery UI for building mobile-friendly sites.

Download jQuery Mobile

When you open the link https://jquerymobile.com/, you will see there are two options to download jQuery mobile library.



Click the *Stable* button, which leads directly to a ZIP file containing the CSS and JQuery files, for the latest version of jQuery mobile library. Extract the ZIP file contents to a jQuery mobile directory.

This version contains all files including all dependencies, a large collection of demos, and even the library's unit test suite. This version is helpful to getting started.

Conclusion: