A Laboratory Manual for

Advanced Web Programming (3151606)

B.E. Semester 6 (Information Technology)

Shantilal Shah
Engineering College

Directorate of Technical Education, Gandhinagar, Gujarat

Shantilal Shah Engineering College, Bhavnagar

Certificate

This is to certify that

Mr./Ms. _____

Enrollment No. _____ of B.E. Semester **6th** Information Technology of this Institute has satisfactorily completed the Practical / Tutorial work for the subject **Advanced Web Programming (3161611)** for the academic year 2023-24.

Place: _____

Date: _____

Name and Sign of Faculty member **Head of the Department**

Preface

Main motto of any laboratory/practical/field work is for enhancing required skills as well as creating ability amongst students to solve real time problem by developing relevant competencies in psychomotor domain. By keeping in view, GTU has designed competency focused outcome-based curriculum for engineering degree programs where sufficient weightage is given to practical work. It shows importance of enhancement of skills amongst the students and it pays attention to utilize every second of time allotted for practical amongst students, instructors and faculty members to achieve relevant outcomes by performing the experiments rather than having merely study type experiments. It is must for effective implementation of competency focused outcome-based curriculum that every practical is keenly designed to serve as a tool to develop and enhance relevant competency required by the various industry among every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual is designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual students can go through the relevant theory and procedure in advance before the actual performance which creates an interest and students can have basic idea prior to performance. This in turn enhances pre-determined outcomes amongst students. Each experiment in this manual begins with competency, industry relevant skills, course outcomes as well as practical outcomes (objectives). The students will also achieve safety and necessary precautions to be taken while performing practical.

This manual also provides guidelines to faculty members to facilitate student centric lab activities through each experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve the outcomes. It also gives an idea that how students will be assessed by providing rubrics.

In the era of digitization, the demand of Internet based applications is increasing day by day. Advanced Web Programming is one of the required skills for IT Engineer. This focuses on front-end and back-end design. After learning this subject students can advance their career in the field of web development.

Utmost care has been taken while preparing this lab manual however always there is chances of improvement. Therefore, we welcome constructive suggestions for improvement and removal of errors if any.

Practical – Course Outcome matrix

Course Outcomes (COs):

CO 1 : Learn the concepts of client side programming using CSS and Java Script

CO 2 : Understand the concepts of Angular JS to extend basic HTML features

CO 3 : Learn Node JS framework to build dynamic server side applications

CO 4 : Study the concept of database using Mongo DB and connect database with application.

CO 5 : Design and implement full featured web application using the concepts of Angular JS and Node JS.

Sr. No.	Objective(s) of Experiment	CO1	CO2	CO3	CO4	CO5
1.	Design a webpage that reads a Text file using AJAX.	√				
2.	Create a HTML form that will accept Enrolment No., Name, Semester, Branch, Mobile Number, Email, Address etc. from the student and display them on the page using Angular JS Directives and Expressions.		√			
3.	Create a webpage using Angular JS that displays details of students' objects (such as Enrolment No., Name, Semester, Branch, Mobile Number, Email Address, Address etc.) in a tabular format with appropriate CSS effects.	√	√			
4.	Modify Practical 3 and provide a search field to search records on the top of the page and also allow user to sort table according to the column values when user clicks on a column using filter.	√	√			
5.	Write Angular JS code to read Customer's data in JSON format available in a Customers.php file using $http service and display the same on a webpage in tabular format.	√	√			
6.	Create an Angular JS application for validation that will accept Email, Username and Password as required fields from the user. It will enable submit button only if all the entered data are valid.		√			
7.	Create an example demonstrating the concept of Angular JS Routing.		√			

8.	Study the installation of Node JS and installation of various packages in Node JS.			√		
9.	Design a webpage with a file input control to browse appropriate file and four buttons Read File Synchronously, Read File Asynchronously, Compress File, Decompress File. Implement the functionality of all four buttons on the browsed file using Node JS.			√		
10.	Create a Node JS application that will allow a user to browse and upload a file in localhost.			√		
11.	Create a Node JS application that will allow a user to create new file, read file, write into a file and delete a file.			√		
12.	Study MongoDB environment setup and write Node JS code to perform insertion operation in Mongo DB.			√	√	
13.	Write Node JS code to perform deletion operation from Mongo DB.			√	√	
14.	Write Node JS code to perform selection and updation operation to select and update specific document in Mongo DB.			√	√	
15.	Create a single page application for Library that will allow the librarian to add a new book and search whether book is currently available in the library or not.	√	√	√	√	√

Industry Relevant Skills

The following industry relevant competency are expected to be developed in the student by undertaking the practical work of this laboratory.

1. HTML/CSS Skills : HTML is used extensively by web developers to build web pages. CSS is used to implement different fonts, colors and layouts in the design of a website.
2. Angular JS Skills : Angular JS is used to create fully functional dynamic web applications.
3. Node JS Skills : Node JS is extensively used to support back end activities in the development of dynamic web applications.
4. Web Development Skills : Angular JS, Node JS are widely used for the development of web applications along with HTML and CSS.

Guidelines for Faculty members

1. Teacher should provide the guideline with demonstration of practical to the students with all features.
2. Teacher shall explain basic concepts/theory related to the experiment to the students before starting of each practical
3. Involve all the students in performance of each experiment.
4. Teacher is expected to share the skills and competencies to be developed in the students and ensure that the respective skills and competencies are developed in the students after the completion of the experimentation.
5. Teachers should give opportunity to students for hands-on experience after the demonstration.
6. Teacher may provide additional knowledge and skills to the students even though not covered in the manual but are expected from the students by concerned industry.
7. Give practical assignment and assess the performance of students based on task assigned to check whether it is as per the instructions or not.
8. Teacher is expected to refer complete curriculum of the course and follow the guidelines for implementation.

Instructions for Students

1. **Students have to write answers / solutions of QUIZ in separate file pages. The quiz of corresponding practical must be attached just behind each practical.**

2. Students are expected to carefully listen to all the theory classes delivered by the faculty members and understand the COs, content of the course, teaching and examination scheme, skill set to be developed etc.
3. Students shall organize the work in the group and make record of all observations.
4. Students shall develop maintenance skill as expected by industries.
5. Student shall attempt to develop related hand-on skills and build confidence.
6. Student shall develop the habits of evolving more ideas, innovations, skills etc. apart from those included in scope of manual.
7. Student shall refer technical magazines and data books.
8. Student should develop a habit of submitting the experimentation work as per the schedule and s/he should be well prepared for the same.

Sample Rubrics for Practical Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Understanding of Problem	Excellent understanding of problem and relevance with the theory clearly understood.	**03**	Moderate level understanding of problem and relevance with the theory clearly understood.	**02**	Problem not understood and can't establish the relation with the theory.	**01**
Implementation of Problem	Efficient implementation with proper naming convention and understanding	**04**	Moderate level of implementation. Poor naming convention.	**03**	Partial implementation with poor understanding.	**01 to 02**
Presentation and report writing	Unique documentation (not copied from other sources) of given problem with proper formatting and language.	**03**	Ordinary documentation of given problem with proper formatting and language	**02**	Weak documentation of given problem without proper formatting and language	**01 to 02**

Index
(Progressive Assessment Sheet)

Sr. No.	Objective(s) of Experiment	Page No.	Date of performance	Date of submission	Assessment Marks	Sign. of Teacher with date	Remarks
1	Design a webpage that reads a Text file using AJAX.						
2	Create a HTML form that will accept Enrolment No., Name, Semester, Branch, Mobile Number, Email, Address etc. from the student and display them on the page using Angular JS Directives and Expressions.						
3	Create a webpage using Angular JS that displays details of students' objects (such as Enrolment No., Name, Semester, Branch, Mobile Number, Email Address, Address etc.) in a tabular format with appropriate CSS effects.						
4	Modify Practical 3 and provide a search field to search records on the top of the page and also allow user to sort table according to the column values when user clicks on a column using filter.						
5	Write Angular JS code to read Customer's data in JSON format available in a Customers.php file using $http service and display the same on a webpage in tabular format.						
6	Create an Angular JS application for validation that will accept Email, Username and Password as required fields from the user. It will enable submit button only if all the entered data are valid.						
7	Create an example demonstrating the concept of Angular JS Routing.						
8	Study the installation of Node JS and installation of various packages in Node JS.						
9	Design a webpage with a file input control to browse appropriate file and four buttons Read File Synchronously, Read File Asynchronously, Compress File, Decompress File. Implement the functionality of all four buttons on the browsed file using Node JS.						
10	Create a Node JS application that will allow a user to browse and upload a file in localhost.						
11	Create a Node JS application that will allow a user to create new file, read file, write into a file and delete a file.						

12	Study MongoDB environment setup and write Node JS code to perform insertion operation in Mongo DB.						
13	Write Node JS code to perform deletion operation from Mongo DB.						
14	Write Node JS code to perform selection and updation operation to select and update specific document in Mongo DB.						
15	Create a single page application for Library that will allow the librarian to add a new book and search whether book is currently available in the library or not.						
Total							

Experiment No: 1

Aim : Design a webpage that reads a Text file using AJAX.

Date :

Competency and Practical Skills: HTML, JavaScript, XMLHttpRequest Object

Relevant CO : CO 1

Objectives:

1. To understand how AJAX can be used to load dynamic content.

Theory:

- **What is AJAX?**
  - AJAX = **A**synchronous **J**avaScript **A**nd **X**ML.
  - AJAX is not a programming language.
  - It just uses a combination of:
    1. A browser built-in XMLHttpRequest object (to request data from a web server)
    2. JavaScript and HTML DOM (to display or use the data)
  - AJAX allows web pages to be updated asynchronously by exchanging data with a web server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.
- **How AJAX works?**

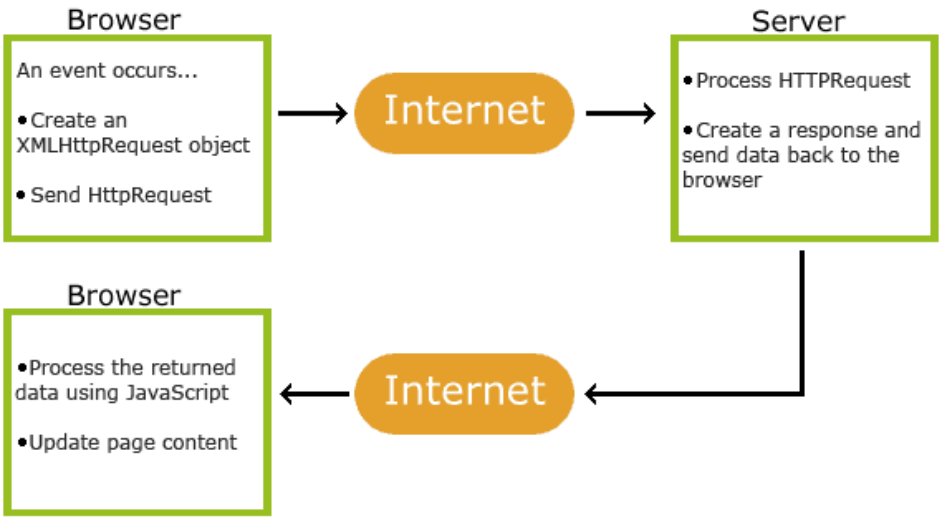

Figure 1. Working of AJAX

  - An event occurs in a web page (the page is loaded, a button is clicked).

- An XMLHttpRequest object is created by JavaScript.
- The XMLHttpRequest object sends a request to a web server.
- The server processes the request.
- The server sends a response back to the web page.
- The response is read by JavaScript.
- Proper action (like page update) is performed by JavaScript.

Implementation : Design a webpage that reads a Text file using AJAX.

Output:

Conclusion:

Quiz:

1. What is AJAX? Enlist the advantages of AJAX.
2. Which type of files can be read using AJAX?

Suggested Reference:

1. https://www.w3schools.com/xml/ajax_intro.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 2

Aim : Create a HTML form that will accept Enrolment No., Name, Semester, Branch, Mobile Number, Email, Address etc. from the student and display them on the page using Angular JS Directives and Expressions.

Date:

Competency and Practical Skills : HTML, Angular JS Directives

Relevant CO : CO 2

Objectives:

1. To understand HTML Page Structure.
2. To understand how to use Angular JS directives.

Theory:

- **HTML Form :** HTML Forms are required, when
  - For example for registration you may collect information like user name , email , contact number, address etc.
  - A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc.
  - The back-end application will perform required processing on the passed data based on defined business logic inside the application.
  - There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.
  - The HTML **<form>** tag is used to create an HTML form

Syntax :

```
<form action = "Script URL" method = "GET|POST">

  form elements like input, textarea etc.

</form>
```

- **Important form attributes are as given below**

Attribute	Description
action	Backend script ready to process your passed data.
method	Method to be used to upload data. The most frequently used are GET and POST methods.
target	Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
enctype	You can use the enctype attribute to specify **how the browser encodes the data before it sends it to the server.** Possible values are – **application/x-www-form-urlencoded** – This is the standard method most forms use in simple scenarios. **mutlipart/form-data** – This is used when you want to upload binary data in the form of files like image, word file etc.

- **HTML Form Controls**

Control Name	Used for	Sample Code
Text input control Single line text input control	Textbox is used for accepting text from user, like firstname, lastname etc	<input type = "text" name = "first_name" />
Password input control	Password input control is used to accept password from user.	< input type = "password" name = "password" />
Text Area Multiline input control.	Teaxtarea is used to accept multiline text input , like comments.	<textarea rows = "5" cols = "50" name = "description"> Enter description here... </textarea>

Checkbox	Checkboxes are used when more than one option is required to be selected.	<input type = "checkbox" name = "maths" value = “maths"> Maths <input type = "checkbox" name = "physics" value = “physics"> name = "password" />
Radio Button	Radio buttons are used when out of many options, just one option is required to be selected.	<input type = "radio" name = "subject" value = "maths"/> Maths <input type = "radio" name = "subject" value = "physics"/> Physics
Drop Down Box	provides option to list down various options in the form of drop down list, from where a user can select one or more options.	<select name = "dropdown"> <option value = "Maths" selected>Maths</option> <option value = "Physics">Physics</option> </select>
File Upload	It allows site users to upload a file to website.it is also known as file select box.	<input type = "file" name = "fileupload" accept = "image/*" />
Submit Button	This creates a button that automatically submits a form.	<input type = "submit" name = "submit" value = "Submit" />

Reset Button	This creates a button that automatically resets form controls to their initial values.	<input type = "reset" name = "reset" value = "Reset" />
Button	This creates a button that is used to trigger a client-side script when the user clicks that button.	<input type = "button" name = "ok" value = "OK" />
Image Button	This creates a clickable button but we can use an image as background of the button.	<input type = "image" name = "imagebutton" src = "/html/images/logo.png" />
Hidden Control	Hidden form controls are used to hide data inside the page which later on can be pushed to the server. This control hides inside the code and does not appear on the actual page.	<input type = "hidden" name = "pagename" value = "10" />

- **Angulr JS Concepts to be used**

  To use Angular JS in web pages, angular.min.js file must be included in the current page using <script src=”angular.min.js” /> tag in head section of the web page.

  - **ng-model :** This directive is used to bind the values of HTML controls to a variable.
  - **ng-bind :** This directive is used to bind HTML elements to variables. It is generally used to display the values of variables in the HTML elements like <p> or .
  - **Angular JS Expression :** Angular JS expressions are used to print the values of the variables on the page. An Angular JS expression starts with {{ and ends with }}.
    - For example, {{ first_name + “ “ + last_name }} will print first_name and last_name separated by space.

Implementation: Create a HTML form that will accept Enrolment No., Name, Semester, Branch, Mobile Number, Email, Address etc. from the student and display them on the page using Angular JS Directives and Expressions.

Output:

Conclusion:

Quiz:

1. Enlist five most commonly used Angular JS directives and state their usages.
2. Explain various ways to display the values of the variables on the page with suitable example.

Suggested References:

1. https://www.w3schools.com/angular/angular_directives.asp
2. https://www.w3schools.com/angular/angular_model.asp
3. https://www.w3schools.com/angular/angular_databinding.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 3

Aim :Create a webpage using Angular JS that displays details of students' objects (such as Enrolment No., Name, Semester, Branch, Mobile Number, Email Address, Address etc.) in a tabular format with appropriate CSS effects.

Date:

Competency and Practical Skills: HTML, CSS, Use of MVC in Angular JS

Relevant CO: CO 1, CO 2

Objectives:

1. To understand about defining objects in Angular JS.
2. To use objects defined in Angular JS and display it's information in HTML table.

Theory:

- **MVC Architecture**

Model **V**iew **C**ontroller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts,

- **Model** − It is the lowest level of the pattern responsible for maintaining data.
- **View** − It is responsible for displaying all or a portion of the data to the user.
- **Controller** − It is a software Code that controls the interactions between the Model and View.

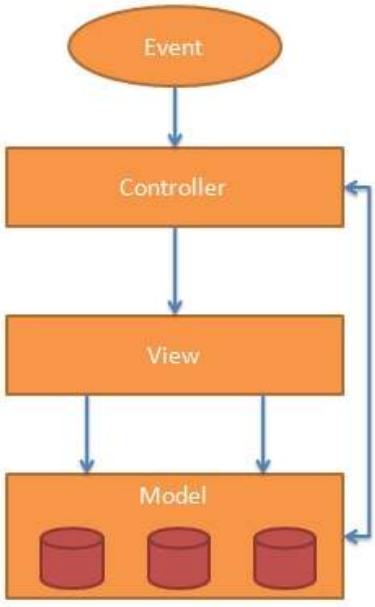

Figure 2. MVC Architecture

MVC is popular because it isolates the application logic from the user interface layer and supports separation of concerns. The controller receives all requests for the application and then works with the model to prepare any data needed by the view. The view then uses the data prepared by the controller to generate a final presentable response. The MVC abstraction can be graphically represented as Figure 2.

- **Angular JS concepts to be used**

  - To define an Angular JS application scope ng-app directive must be used.
  - After defining an Angular JS application, it can be bind with a controller using ng-controller directive.
  - To define controller, first an application module must be defined in <script> tag using

    ```
    var app = angular.module('myApp', [ ]);
    ```

    Here **angular.module ( )** function is used to define application module. Where **myApp** is the name of application and **[ ]** (empty square brackets) are used to mention no additional / dependant modules are needed / used.

  - Then using **app** variable controller can be defined as

    ```
    app.controller('myCtrl', function($scope) {
        $scope.students=[
          {enrolment:'301',name:'Rajesh',semester:'3',branch:'I.T.'},
          {enrolment:'501',name:'Ishwar',semester:'5',branch:'Production'},
          {enrolment:'701',name:'Ronit',semester:'7',branch:'Mechanical'}];
    });
    ```

    Application controller is defined using **app** variable along with **controller( )** function. Controller( ) function takes two arguments, first is the name of controller, which is **myCtrl** in our case and second is the callback function with a single variable that is **$scope**. $scope is an object, that can be used to declare variables and objects within the controller and it is accessible throughout the controller.

    As it can be seen in the above code, **students** is an array of objects defined using $scope. Each object contains various values including **enrolment**, **name**, **semester** and **branch**.

  - A table must be created using <table> tag.
  - Now to create separate rows for each objects in a table, the following syntax cab be used.

    ```
    <tr ng-repeat="x in students">
    ```

    Here, **ng-repeat** directive will iterate over all the elements of **students** (array of objects) and separate rows will be created for each object. Each time object will be accessed using

x and you will be able to print it's variables using expression, such as, **{{ x.enrolment }}** in individual cells of a table row.

- **HTML Table Tag**
  - HTML tables allow web developers to arrange data into rows and columns.
  - The **<table>** tag defines an HTML table.
  - table row is defined with a **<tr>** tag.
  - table header is defined with a **<th>** tag.
  - text in **<th>** elements are bold and centered.
  - Each table data/cell is defined with a **<td>.**
  - By default, the text in **<td>** elements are regular and left-aligned.
  - **colspan** attribute is used to make a cell span more than one column.
  - **rowspan** attriute is used to make a call span more than one row.
  - **cellpadding** represents the distance between cell borders and the content within a cell.
  - The **cellspacing** attribute defines space between table cell

Implementation: Create your class time table using table tag, experiment with rowspan, colspan, cellspacing and cellpadding attributes.

Output:

Conclusion:

Quiz:

1. Write a note on : MVC Architecture.
2. Explain the usage of ng-repeat, ng-controller directives with suitable example.

Suggested Reference:

1. https://www.w3schools.com/angular/angular_directives.asp
2. https://www.w3schools.com/angular/angular_tables.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 4

Aim : Modify Practical 3 and provide a search field to search records on the top of the page and also allow user to sort table according to the column values when user clicks on a column using filter.

Date:

Competency and Practical Skills: HTML, CSS, MVC in Angular JS, Filters in Angular JS

Relevant CO: CO 1, CO 2

Objectives:

1. To understand about defining objects in Angular JS.
2. To use objects defined in Angular JS and display it's information in HTML table.
3. To use filter to filter the records.

Theory:

- **Angular JS concepts to be used**
  - **Filters in Angular JS :** Filters in Angular JS are used to format / transform the data. Angular JS provides the following filters.
    - **currency** : It is used to format a number to a currency format.
    - **date** : It is used to format a date to a specified format.
    - **filter** : It is used to select a subset of items from an array.
    - **json** : It formats an object to a JSON string.
    - **limitTo** : It limits an array/string, into a specified number of elements/characters.
    - **lowercase** : It is used to format a string to lower case.
    - **number** : It formats a number to a string.
    - **orderBy** : It orders / sorts an array by an expression.
    - **uppercase** : It is used to format a string to upper case.

    For example,

```
<ul>
        <li ng-repeat="x in names | filter : 'i'">
                {{ x }}
        </li>
</ul>
```

The above code will display an unordered list of items from **names** array consisting **'i**' as a character.

```
<ul>
        <li ng-repeat="x in persons | orderBy:'country'">
                {{ x.name + ', ' + x.country }}
        </li>
</ul>
```

Similarly the above code will display the **name** and **country** of all the elements of **persons** array separated by comma in an unordered list, and this list will be sorted by value of **country** variable of each object.

- **Defining functions in Angular JS**

In Angular JS functions can be defined inside the controller using **$scope** object. The general syntax to define a function is,

```
$scope.function_name = function(value1, value2, ...) {
                //body of the function
}
```

Implementation: Modify Practical 3 and provide a search field to search records on the top of the page and also allow user to sort table according to the column values when user clicks on a column using filter.

Output:

Conclusion:

Quiz:

1. Discuss various filters present in Angular JS with suitable examples.
2. Write Angular JS code to display numbers in Rupee format with Rupee symbol.

Suggested Reference:

1. https://www.w3schools.com/angular/angular_filters.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 5

Aim : Write Angular JS code to read Customer's data in JSON format available in a Customers.php file using $http service and display the same on a webpage in tabular format.

Date:

Competency and Practical Skills: HTML, JSON (JavaScript), PHP, Services in Angular JS

Relevant CO: CO 1, CO 2

Objectives:

1. To understand the use of services in Angular JS.

Theory:

- **Angular JS concepts to be used**
  - **Services in Anguar JS :** In AngularJS, a service is a function, or object, that is available for, and limited to, your AngularJS application. AngularJS has more than 30 built-in services. Some of the most commonly used services are,
    - **$location :** It is used to get the location of the current webpage (URL).
    - **$http :** It is used to request the resources available on the server and lets your application handle the response.
    - **$timeout :** It is similar to setTimeout( ) function of JavaScript. It is used to perform an activity when time outs.
    - **$interval :** It is similar to setInterval( ) function of JavaScript. It is used to perform some task repeatedly at some interval.
  - **$http Service :** The AngularJS $http service makes a request to the server, and returns a response. The following code demonstrates the general use of $http service.

```
<script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $http) {
  $http.get("MyPage.php").then(function(response) {
    $scope.myData = response.data.records;
  });
});
</script>
```

Here an object of **$http** service needs to be passed in the callback function of controller. Which can further be used to request the resources available on the server. Here it tries to

access **MyPage.php**. If resource request is handled then the response will be available in **response** variable and that can be used to further process the output as per the user's need.

Implementation: Write Angular JS code to read Customer's data in JSON format available in a Customers.php file using $http service and display the same on a webpage in tabular format.

Output:

Conclusion:

Quiz:

1. Explain the usage of any five services available in Angular JS with suitable examples.
2. Write Angular JS code to display clock on the web page.

Suggested Reference:

1. https://www.w3schools.com/angular/angular_services.asp
2. https://www.w3schools.com/angular/angular_http.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 6

Aim : Create an Angular JS application for validation that will accept Email, Username and Password as required fields from the user. It will enable submit button only if all the entered data are valid.

Date:

Competency and Practical Skills: HTML, Validation states / flags Angular JS

Relevant CO: CO 2

Objectives:

1. To understand the use of validation states / flags available in Angular JS for validating data captured from the users

Theory:

- **Angular JS concepts to be used**
  - **Validation states / flags in Anguar JS :** Angular JS provides facility to observe the changes made in the form controls through various states / flags. Some of the states are related to input controls and some can be used for forms.
    - **Input fields have the following states:**
      1. **$untouched** : It is used to identify that the field has not been touched yet.
      2. **$touched** : It is used to identify that the field has been touched.
      3. **$pristine** : It is used to identify that the field has not been modified yet.
      4. **$dirty** : It is used to identify that the field has been modified.
      5. **$invalid** : It is used to identify that the field content is not valid.
      6. **$valid** : It is used to identify that the field content is valid.

      They are all properties of the input field, and are either true or false.

    - **Forms have the following states:**
      1. **$pristine** : It is used to identify that no fields have been modified yet.
      2. **$dirty** : It is used to identify that one or more fields of the form have been modified.
      3. **$invalid** : It is used to identify that the form content is not valid.
      4. **$valid** : It is used to identify that the form content is valid.
      5. **$submitted** : It is used to identify that the form is submitted

      They are all properties of the form, and are either true or false.

```
<body ng-app="">
        <form name="Form1">
        Name: <input name="Name" ng-model="Name" required>
        <span ng-show="Form1.Name.$touched && myForm.myName.$invalid">The
        name is required.</span>
        </form>
</body>
```

The above code shows an example of using these states / flags. In this code, an input field **Name** of **Form1** is validated using **Form1.Name.$touched** and **myForm.myName.$invalid**. If the input control is modified then the value of **Form1.Name.$touched** will be true, otherwise false. Similarly if the input control's content is empty then the value of **myForm.myName.$invalid** will be true, otherwise false.

Here, **ng-show** directive is used to show / hide the element based on the result of the flags. Thus, element will be visible only if both the states / flags are true, means the input control is modified and empty.

Similarly **ng-enabled** and **ng-disabled** directives can be used to enable or disable the controls in the form. Both of these are assigned boolean values.

Implementation: Create an Angular JS application for validation that will accept Email, Username and Password as required fields from the user. It will enable submit button only if all the entered data are valid.

Output:

Conclusion:

Quiz:

1. Briefly discuss the usage of all the Input states and Form states with suitable examples.
2. Discuss various CSS classes present in Angular JS used for these validation states.

Suggested Reference:

1. https://www.w3schools.com/angular/angular_validation.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 7

Aim : Create an example demonstrating the concept of Angular JS Routing.

Date:

Competency and Practical Skills: HTML, Routing in Angular JS

Relevant CO: CO 2

Objectives:

1. To understand how to create Single Page Application (SPA) using Angular JS Routing.

Theory:

- **Angular JS concepts to be used**
  - **Routing in Anguar JS :** In Angular JS, **ngRoute** module helps in configuring routing information to create Single Page Applications.

    In order to use ngRoute module, first the developer need to include **angular-route.js** file using the following line in <head> section.

```
<script src=”angular-route.js” />
```

    In the second step, links to load the pages needs to be created. For example,

```
<a href="#!red">Red</a>
```

    In the third step, **ngRoute** module must be included as a dependency in application module using the following line,

```
var app = angular.module("myApp", ["ngRoute"]);
```

    Now your application has access to the route module, which provides the **$routeProvider** object. In the fourth step, use this **$routeProvider** object to configure routing details in the Angular JS application. The following code demonstrates the same,

```
app.config(function($routeProvider) {
 $routeProvider
 .when("/", {
  templateUrl : "main.htm"
 })
 .when("/red", {
  templateUrl : "red.htm"
 })
 .when("/green", {
  templateUrl : "green.htm"
 })
 .when("/blue", {
  templateUrl : "blue.htm"
```

```
  });
});
```

In the above code, in **when( )** function, two arguments needs to be passed, first is the target URL and second is the block of statements to execute when target URL is requested. Here, **templateUrl** is used to load HTML page as a template.

Now to load the content of resource specified by **templateUrl**, a view needs to be created. This can be done using **ng-view** directive in three different ways.

1. Using **ng-view** directive can be used as HTML element.

```
<ng-view></ng-view>
```

2. Using **ng-view** directive as an attribute in HTML element.

```
<div ng-view></div>
```

3. Using ng-view as a value to the class attribute in HTML element.

```
<div class="ng-view"></div>
```

Once, the view is specified, the pages will be loaded in the view when the links are clicked.

Implementation: Create an example demonstrating the concept of Angular JS Routing.

Output:

Conclusion:

Quiz:

1. Write a note on : Routing in Angular JS.
2. State the usefulness of ng-view directive. Explain various ways to use it.

Suggested Reference:

1. https://www.w3schools.com/angular/angular_routing.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 8

Aim : Study the installation of Node JS and installation of various packages in Node JS.

Date:

Competency and Practical Skills: Node JS Environment setup

Relevant CO: CO 3

Objectives:

1. To learn installation of Node JS and the required packages in Node JS.

Theory:

- **Node JS Installation**

First, Node JS installation package needs to be downloaded from,

https://nodejs.org/en

After downloading the package, double click on it to install Node JS in your machine. Follow the instructions one by one and act accordingly. This will install Node JS with some basic packages such as, fs, os, net, dns, url, http, https, path etc. in your machine. In order to install additional packages **Node Package Manager (NPM)** can be used.

- **Node Package Manager**

Node Package Manager allows us to install new packages, update existing packages and uninstall existing packages. All can be done by executing following commands on Command Line Interface provided by the operating system. Lets have a look at some most commonly used commands.

- To install new package locally, use,

  npm install package_name
- To install new package globally, use,

  npm install -g package_name

  Here -g flag is used to install package globally
- To update existing package, use,

  npm update package_name
- To uninstall already installed package, use,

  npm uninstall package_name
- To get a list of locally installed packages, use

  npm list
- To get a list of locally installed packages, use

npm list -g

Output:

<Include the Screenshots of installing Node JS environment>

<Include the Screenshots of managing packages using NPM>

Conclusion:

Quiz:

1. Write a note on : Routing in Angular JS.
2. State the usefulness of ng-view directive. Explain various ways to use it.

Suggested Reference:

1. https://www.geeksforgeeks.org/15-npm-commands-that-every-node-js-developer-should-know/

References used by the students:

Assessment :

Node JS installation process (4 marks)	Understanding of NPM (3 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Node JS installation process	Detailed screenshots are given, which are sufficient to understand the process	**04**	Required screenshots are given, which can be used to understand the process	**03**	Partial screenshots are given, which are nor sufficient to understand the process.	**01 to 02**
Understanding of NPM	All the commands are given with detailed information and output.	**03**	Sufficient commands are given with required information and output.	**02**	Partial commands are discussed with or without output.	**01**
Presentation and report writing	Unique documentation (not copied from other sources) of given problem with proper formatting and language.	**03**	Ordinary documentation of given problem with proper formatting and language	**02**	Weak documentation of given problem without proper formatting and language	**01 to 02**

Experiment No: 9

Aim : Design a webpage with a file input control to browse appropriate file and four buttons Read File Synchronously, Read File Asynchronously, Compress File, Decompress File. Implement the functionality of all four buttons on the browsed file using Node JS.

Date :

Competency and Practical Skills: File System (fs) module and zlib modules of Node JS

Relevant CO : CO 3

Objectives:

1. To understand to difference between synchronous and asynchronous processes.
2. To understand the functions for compressing and decompressing files and use them.

Theory:

- **Node JS concepts to be used**

  Node JS provides variety of functions to perform various operations on files. To use file system related functions, file system module must be included in the current code using require( ) function. For example,

  var fs = require("fs");

  Now using fs variable, all the functions defined in file system module can be accessed.

  - **Reading a file synchronously**

    To read a file synchronously **readFileSync( )** function is used. It takes only one parameter, that is the URL / Name of the file (along with the extension). To display read data in appropriate format, you need to convert it in specific types. When a file is being read synchronously, other processes will be blocked till the reading is done.

  - **Reading a file asynchronously**

    To read a file asynchronously **readFile( )** function is used. It takes two parameters, first is the URL / Name of the file (along with the extension) and second is the callback function. To display read data in appropriate format, you need to convert it in specific types. When a file is being read asynchronously, other processes will continue their execution in parallel while reading is done.

  - **Compressing a file**

    To compress a file first of all you need to read the file. **createReadStream( )** function of **file system module** allows you to read a file. It takes a single parameter that is the URL / Name of the file to be read (along with the extension).

    After reading a file you can compress it using **zlib** module's **createGzip( )** function.

Then you need to write the compressed file, that can be done using **createWriteStream( )** function of **file system module**, again this function takes a single parameter, that is the name of the file to be saved with.

In order to combine these function **pipe( )** function can be used. It allows us to provide output of one function as an input to the other one.

- **Decompressing a file**

  Similarly to decompress a file you need to read the file using the same function previously used **createReadStream( )**.

  To decompress a file , use **createGunzip( )** function of **zlib** module.

  And again use **createWriteStream( )** function to write a decompressed file.

Implementation : Design a webpage with a file input control to browse appropriate file and four buttons Read File Synchronously, Read File Asynchronously, Compress File, Decompress File. Implement the functionality of all four buttons on the browsed file using Node JS.

Output:

Conclusion:

Quiz:

1. Enlist various functions available in **fs** module and also state their usages.
2. Explain various functions available in **zlib** module.

Suggested Reference:

1. https://www.knowledgehut.com/blog/web-development/compression-decompression-of-data-using-zlib-in-Nodejs#there-are-two-types-of-compression:%E2%80%AFlossless%E2%80%AFand%E2%80%AFlossy.

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 10

Aim : Create a Node JS application that will allow a user to browse and upload a file in localhost.

Date :

Competency and Practical Skills: File System (fs), http, and formidable modules of Node JS

Relevant CO : CO 3

Objectives:

1. To understand form processing while uploading any file on server in Node JS.

Theory:

- **Node JS concepts to be used**

To process the form, formidable module is required. It is not included in the basic Node JS installation. Developers can install the formidable module using NPM through the following command,

```
npm install formidable
```

In addition to formidable, file system (fs) and http modules will be required.

While designing the interface that will allow a user to browse the file to be uploaded, one important thing to remember is to use **post** as a method and specify **enctype** attribute in <form> tag, like,

```
<form action="URL of the page" method="post" enctype="multipart/form-data">
```

After these modifications, your form will be able to upload the files on the server.

Now as shown bellow, using the **IncomingForm( )** method of **formidable** module, the form can be referred using a variable.

```
var form = new formidable.IncomingForm();
```

After that form variable can be used to parse the request object, fields and files using **parse( )** function. Using appropriate methods of file system (fs) module, the uploaded file can be stored in specific directory with appropriate name.

Implementation : Design a webpage with a file input control to browse appropriate file and four buttons Read File Synchronously, Read File Asynchronously, Compress File, Decompress File. Implement the functionality of all four buttons on the browsed file using Node JS.

Output:

Conclusion:

Quiz:

1. Enlist and explain various methods of formidable module used to manage uploading files.

Suggested Reference:

1. https://www.w3schools.com/nodejs/nodejs_uploadfiles.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 11

Aim : Create a Node JS application that will allow a user to create new file, read file, write into a file and delete a file.

Date :

Competency and Practical Skills: File System (fs) module of Node JS

Relevant CO : CO 3

Objectives:

1. To understand the basics of file management in Node JS.

Theory:

- **Node JS concepts to be used**

To manage files using Node JS applications, file system (fs) module is required. It provides various functions to manage the files.

Following functions will be used to manage files using Node JS :

1. **readFile( )** : It is used to read a file.
2. **appendFile( )** : It is used to append content in a file. If file doesn't exists, new file will be created.
3. **open( )** : If open( ) is used with 'w' flag, it will allow user to write into a file. If file doesn't exists empty file will be created.
4. **writeFile( )** : It is used to write a file. If file doesn't exists, new file will be created.
5. **unlink( )** : It is used to delete a file.
6. **rename( )** : It is used to rename an existing file.

Implementation : Create a Node JS application that will allow a user to create new file, read file, write into a file and delete a file.

Output:

Conclusion:

Practical Quiz:

1. Create Node JS application that will list all the files available in the browsed directory of a server.
2. Create Node JS application that will allow a user to create new files and rename existing files using the proper interface

Suggested Reference:

1. https://www.w3schools.com/nodejs/nodejs_filesystem.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 12

Aim : Study MongoDB environment setup and write Node JS code to perform insertion operation in Mongo DB.

Date :

Competency and Practical Skills: Operations on MongoDB Database, Node JS code for insertion in MongoDB Database.

Relevant CO : CO 3, CO 4

Objectives:

1. To understand the installation of MongoDB.
2. To understand how to interact with MongoDB Database through Node JS code.

Theory:

- **Introduction to MongoDB**

  MongoDB is a document database which is often referred to as a non-relational database. This does not mean that relational data cannot be stored in document databases. It means that relational data is stored differently. A better way to refer to it is as a non-tabular database.

  MongoDB stores data in flexible documents. Instead of having multiple tables you can simply keep all of your related data together. This makes reading your data very fast.

  You can still have multiple groups of data too. In MongoDB, instead of tables these are called collections.

  MongoDB can be used locally or you can use the cloud platform for the same.

- **Node JS concepts to be used**
  - After successful installation of MongoDB, services of MongoDB will be accessible through **MongoClient** object of **mongodb** package in Node JS.
  - **connect( )** function of **MongoClient** will allow a user to create a database, if database doesn't exists.
  - All the documents of a MongoDB database are maintained in a collection, a collection is similar to a table in relational databases.
  - To create a collection, **createCollection( )** function of Database object can be used.
  - And **collection( )** function of Database object can be used to access the collection.
  - In MongoDB the records are inserted in terms of documents, therefore the object of document(s) to be inserted must be defined before performing insertion operation.
  - To insert a single document **insertOne( )** function is used.

- In case of insertion of multiple documents, **insertMany( )** function can be used.

Implementation : Study MongoDB environment setup and write Node JS code to perform insertion operation in Mongo DB.

Output:

Conclusion:

Practical Quiz:

1. Differentiate between Relational Database and Document Database. **OR** Differentiate between SQL Database and No SQL Database.
2. Discuss various methods available in Node JS to perform various operations on MongoDB.

Suggested Reference:

1. https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/
2. https://www.w3schools.com/nodejs/nodejs_mongodb_insert.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 13

Aim : Write Node JS code to perform deletion operation from Mongo DB.

Date :

Competency and Practical Skills: Operations on MongoDB Database, Node JS code for deletion in MongoDB Database.

Relevant CO : CO 3, CO 4

Objectives:

1. To understand how to delete documents from a MongoDB Database through Node JS code.

Theory:

- **Node JS concepts to be used**
  - Services of MongoDB will be accessible through **MongoClient** object of **mongodb** package in Node JS.
  - **connect( )** function of **MongoClient** will allow a user to connect with a MongoDB database.
  - Once the connection with the Database is established, **collection( )** function of Database object can be used to access the collection.
  - After that create an object containing query data for a document to be deleted.
  - The use **deleteOne( )** function of collection to delete single document.
  - You can use **deleteMany( )** function of collection to delete multiple documents simultaneously.

Implementation : Write Node JS code to perform deletion operation from Mongo DB.

Output:

Conclusion:

Suggested Reference:

1. https://www.w3schools.com/nodejs/nodejs_mongodb_query.asp
2. https://www.w3schools.com/nodejs/nodejs_mongodb_delete.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 14

Aim : Write Node JS code to perform selection and updation operation to select and update specific document in Mongo DB.

Date :

Competency and Practical Skills: Operations on MongoDB Database, Node JS code for selection and updation of document in MongoDB Database.

Relevant CO : CO 3, CO 4

Objectives:

1. To understand how to search documents from a MongoDB Database and updated them through Node JS code.

Theory:

- **Node JS concepts to be used**
  - Services of MongoDB will be accessible through **MongoClient** object of **mongodb** package in Node JS.
  - **connect( )** function of **MongoClient** will allow a user to connect with the MongoDB database.
  - Once the connection with the Database is established, **collection( )** function of Database object can be used to access the collection.
  - After that object containing query data for a document to be selected needs to be created.
  - Using **find( )** method of the collection object, document can be searched.
  - If the document is found then it's details can be printed / used as per need.
  - For updating documents, use **updateOne( )** function of collection to update a single document and use **updateMany( )** function of collection, to update multiple documents.

Implementation : Write Node JS code to perform selection and updation operation to select and update specific document in Mongo DB.

Output:

Conclusion:

Suggested Reference:

1. https://www.w3schools.com/nodejs/nodejs_mongodb_query.asp
2. https://www.w3schools.com/nodejs/nodejs_mongodb_update.asp

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)

Experiment No: 15

Aim : Create a single page application for Library that will allow the librarian to add a new book and search whether book is currently available in the library or not.

Date :

Competency and Practical Skills: HTML, CSS, Angular JS, Node JS, MongoDB.

Relevant CO : CO 1, CO 2, CO 3, CO 4, CO 5

Objectives:

1. To learn application development using HTML, CSS, Angular JS and Node JS.

Implementation : Create a single page application for Library that will allow the librarian to add a new book and search whether book is currently available in the library or not.

Output:

Conclusion:

Suggested Reference:

1. https://www.w3schools.com/html/default.asp
2. https://www.w3schools.com/css/default.asp
3. https://www.w3schools.com/angular/

4. https://www.w3schools.com/nodejs/
5. https://www.w3schools.com/mongodb/index.php

References used by the students:

Assessment :

Understanding of Problem (3 marks)	Implementation of Problem (4 marks)	Presentation and report writing (3 marks)	Total (10 marks)