A PROJECT REPORT

On

DIGITAL LIBRARY AND BOOK RECOMMENDATION

Submitted in partial fulfillment of the requirement of
University of Mumbai for

Internet Programming Mini Project
In
Information Technology

Submitted By
Shravani Patil
Samarth Patil
Sameer Shahi

Department Of Information Technology
PILLAI COLLEGE OF ENGINEERING
New Panvel – 410 206
UNIVERSITY OF MUMBAI

Academic Year 2023 – 24

DEPARTMENT OF INFORMATION TECHNOLOGY
Pillai College of Engineering
New Panvel – 410 206

CERTIFICATE

This is to certify that the requirements for the report entitled '**Digital Library and Book Recommendation**' have been successfully completed by the following students:

Name	**Roll No.**
Shravani Patil	B548
Samarth Patil	B547
Sameer Shahi	B553

in partial fulfillment of Internet Programming mini Project in the Department of Information Technology, Pillai College of Engineering, New Panvel – 410 206 during the Academic Year 2023 – 2024.

_____

Prof. Prerana Kulkarni

DEPARTMENT OF INFORMATION TECHNOLOGY
Pillai College of Engineering
New Panvel – 410 206

PROJECT APPROVAL FOR

This project entitled "**Digital Library and Book Recommendation**" by Shravani Patil, Samarth Patil, Sameer Shahi are approved for the degree of Bachelor of Engineering in Information Technology.

Examiners:

1. _____

2. _____

3. _____

4. _____

5. _____

Date:

DEPARTMENT OF INFORMATION TECHNOLOGY
Pillai College of Engineering
New Panvel – 410 206

DECLARATION

We declare that this written submission for the Internet Programming Mini Project entitled "**Digital Library and Book Recommendation**" represents our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any ideas / data / fact / source in our submission. We understand that any violation of the above will cause disciplinary action by the institute and also evoke penal action from the sources which have not been properly cited or from whom prior permission have not been taken when needed.

Project Group Members:

Shravani Patil & Sign:

_____

Samarth Patil & Sign:

_____

Sameer Shahi & Sign:

_____

Abstract

In today's digital world, traditional libraries are transforming into digital libraries. These digital libraries are like vast online book collections that you can access from anywhere. Alongside this, there are smart systems that recommend books based on what you like to read. This paper explores how these digital libraries and book recommendation systems work together. Digital libraries use technology to organize and share books, making them available on computers and phones. Recommendation systems use clever algorithms to suggest books you might enjoy based on your interests. We'll look at how these systems are set up, how they make suggestions, and the challenges they face. By understanding these digital tools, we can see how they're changing the way we discover and enjoy books in the digital age. Imagine a library where you can explore countless books without leaving your home. That's the magic of digital libraries! This paper is like a guide to this digital wonderland and its sidekick – book recommendation systems. Digital libraries are like huge online book collections. They use cool tech to organize books so you can find them easily on your computer or phone. But it doesn't stop there! We also have these smart systems that suggest books you might love, just like a helpful friend who knows your taste in stories. We'll dig into how digital libraries and recommendation systems team up. Think of it like a behind-the-scenes tour. We'll see how these systems know what books you might enjoy, how they make suggestions, and the tricky parts they need to figure out. Join the adventure as we explore the tech that's changing how we find and fall in love with books in this digital era!

Table of Contents

List of Figures

Chapter 1
Introduction

In the fast-evolving landscape of information management and technological advancement, traditional libraries are undergoing a transformative shift toward digital platforms. The advent of E-Libraries has not only revolutionized the way books are accessed but has also opened up new possibilities for efficient management and personalized user experiences. This report explores the dynamic realm of E-Library Management and its integration with cutting-edge Book Recommendation Systems, aiming to enhance accessibility, convenience, and engagement for library patrons. The traditional library paradigm, with its physical shelves and catalog systems, is gradually being replaced by virtual repositories of knowledge. E-Library Management involves the digitization and organization of vast collections, providing users with seamless access to a plethora of resources. This shift not only overcomes the limitations of physical space but also facilitates global accessibility, fostering a borderless intellectual community.

Transitioning from physical to digital cataloging enables efficient storage, retrieval, and organization of resources, allowing users to locate information with unprecedented speed. Implementing secure authentication mechanisms ensures that only authorized users can access copyrighted material, protecting intellectual property while promoting responsible usage. E-Libraries are designed to be accessible across various devices, making knowledge available to users wherever and whenever they need it, whether on a computer, tablet, or smartphone.

While E-Libraries offer an abundance of resources, navigating through extensive collections can be overwhelming. Here, Book Recommendation Systems play a pivotal role in enhancing the user experience by providing personalized suggestions tailored to individual preferences. By analyzing user behavior, preferences, and historical interactions, these systems create personalized profiles, ensuring recommendations align with individual tastes and interests. Leveraging collective user behavior data, collaborative filtering algorithms identify patterns and similarities among users, recommending items based on the preferences of similar users. Examining the content of books

and user profiles, content-based filtering suggests items that match users' past preferences, ensuring recommendations are relevant and aligned with individual tastes. This report aims to provide a comprehensive understanding of the integration of E-Library Management and Book Recommendation Systems. By exploring the benefits, challenges, and emerging trends in these fields, we seek to empower libraries and educational institutions to make informed decisions in adopting and optimizing digital solutions. The subsequent sections will delve into each aspect, offering insights into the technologies, best practices, and potential future developments in the quest to create efficient, user-centric digital libraries.

Chapter 2

Requirement Analysis

Requirement analysis for a web application project is a critical step in its development. It involves understanding and documenting the needs, goals, and constraints of the project. Requirement analysis techniques play a very important role in any of your web or mobile app development project's success.

Functional Requirements

Functional requirements for a digital library and book recommendation web app specify the features and capabilities that the application should have. Below are key functional requirements for such an app :

1. **User Registration and Authentication:**
    a. Users can create accounts and log in securely.
    b. Users can reset or recover their passwords.
2. **Search and Browse:**
    a. Users can search for books by title, author, genre, ISBN, or keywords.
    b. Browse books by categories, authors, and new arrivals.
3. **Book Details:**
    a. Users can view detailed information about each book, including the title, author, description, cover image, publication date.
4. **Recommendation Engine:**
    a. Provide personalized book recommendations based on user behavior, ratings, and reading history.
5. **Mobile Responsiveness:**
    a. The web app should be responsive and function well on various devices and screen sizes.

Non Functional Requirements

Non-functional requirements for a digital library and book recommendation web app describe the quality attributes and constraints that are essential for the application's success. These non-functional requirements are just as critical as the functional ones. Here are some important non-functional requirements :

1. **Performance:**
    a. The system should respond quickly to user requests, with fast page loading times and minimal latency.
2. **Scalability:**
    a. The application must handle an increasing number of users and books without significant degradation in performance. It should scale horizontally or vertically as needed.
3. **Reliability:**
    a. The system should be highly available and reliable, with minimal downtime for maintenance or updates.
4. **Usability:**
    a. The user interface should be intuitive and user-friendly, ensuring easy navigation and readability. It should follow best practices for user experience (UX) design.
5. **Security:**
    a. User data and transactions should be securely encrypted.
    b. Implement security measures to protect against common web vulnerabilities, such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).

Chapter 3

Wireframe

A website wireframe, also known as a page schematic or screen blueprint, is a visual guide that represents the skeletal framework of a website. Wireframes are created for the purpose of arranging elements to best accomplish a particular purpose. The purpose is usually being informed by a business objective and a creative idea. The wireframe depicts the page layout or arrangement of the website's content, including interface elements and navigational systems, and how they work together. The wireframe usually lacks typographic style, color, or graphics, since the main focus lies in functionality, behavior, and priority of content. In other words, it focuses on what a screen does, not what it looks like. Wireframes can be pencil drawings or sketches on a whiteboard, or they can be produced by means of a broad array of free or commercial software applications. Wireframes are generally created by business analysts, user experience designers, developers, visual designers, and by those with expertise in interaction design, information architecture and user research.

Wireframes focus on:

1. The range of functions available
2. The relative priorities of the information and functions
3. The rules for displaying certain kinds of information
4. The effect of different scenarios on the display

Wireframe:-

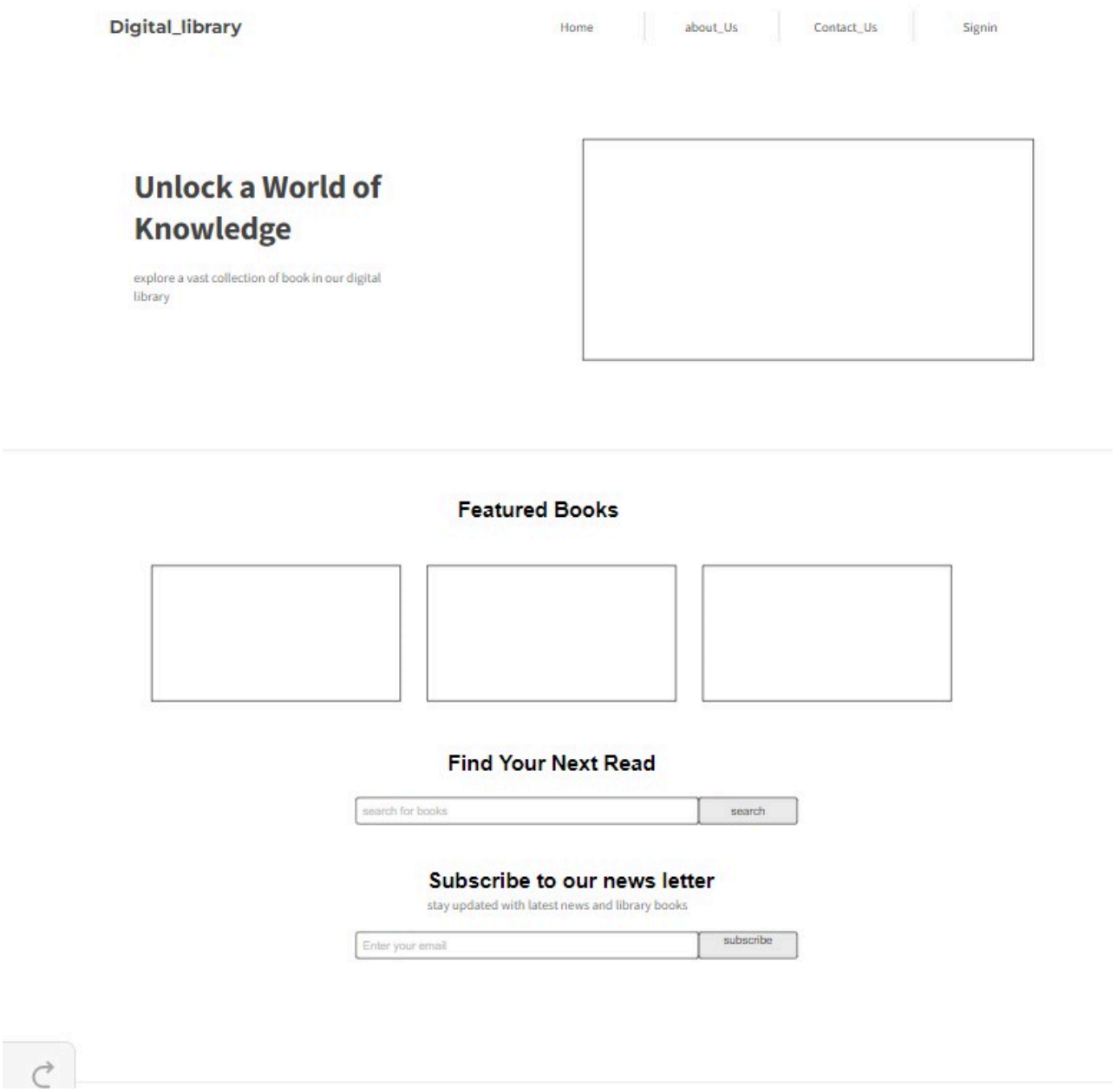


Fig 3.1: Wireframe

Chapter 4

Using different types of CSS

Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents. CSS is a language that describes the style of an HTML document. CSS describes how HTML elements should be displayed.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C).

Syntax of CSS

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts :

Selector − A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.

Property - A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be color, border etc.

Value - Values are assigned to properties. For example, color property can have value either red or #F1F1F1 etc.

selector { property: value } selector { property: value, property: value }
h1 {‘color’:’blue’}

Types of css selectors:

- The element Selector :h1 {color: red;}
- The id Selector / type selector :#para1 {text-align: center;color: red;}
- The Descendant Selectors : ul li {color: red;}
- The class Selector : .center {text-align: center;color: red;}
- The Attribute Selectors : input[type = "text"]{color: #000000; }
- The Child Selectors : body > p {color: #000000; }
- The Universal Selectors : * {color: red;}
- The Adjacent Sibling Selector : H2+P {color: red;}

There are three ways of inserting a style sheet:

1. External style sheet
2. Internal style sheet
3. Inline style

External style sheet

The <link> element can be used to include an external stylesheet file in your HTML document. An external style sheet is a separate text file with .css extension. You define all the Style rules within this text file and then you can include this file in any HTML document using <link> element.

Here is the generic syntax of including external CSS file −

```
<head>
   <link type = "text/css" href = "..." media = "..." />
</head>
```

Consider a simple style sheet file with a name mystyle.css having the following rules −

```
h1, h2, h3 {
   color: #36C;
   font-weight: normal;
   letter-spacing: .4em;
   margin-bottom: 1em;
   text-transform: lowercase;
}
```

Now you can include this file mystyle.css in any HTML document as follows −

```
<head>
   <link type = "text/css" href = "mystyle.css" media = " all" />
</head>
```

Internal style sheet

You can put your CSS rules into an HTML document using the <style> element. This tag is placed inside <head>...</head> tags. Rules defined using this syntax will be applied to all the elements available in the document

<style> Attribute type =”text/css” <style type=”text/css” > </style>

Specifies the style sheet language as a content-type (MIME type). This is required attribute.

media attribute <style type = "text/css" media = "all">

```
<head>
<style type=”text/css” >
body {
        background-color: linen;
}
h1 {
        color: maroon;
        margin-left: 40px;
        }
</style>
</head>
```

Inline style

An inline style may be used to apply a unique style for a single element. To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property. Syntax: <element style = "...style rules...."">

Attributes style “style rules”

The value of style attribute is a combination of style declarations separated by semicolon (;)

<h1 style = "color:#36C;">

Here we have applied the external css for the body element

Code:-

```
element.style {
    background-image: linear-gradient(90deg,#a6a6a6,#ffffff);
}
body {
    margin: 0;
    font-size: 1rem;
    font-weight: 400;
    line-height: 1.5;
    color: #212529;
    text-align: left;
    background-color: #fff;
}

*, ::after, ::before {
    box-sizing: border-box;
}
user agent stylesheet
body {
    display: block;
    margin: 8px;
}

*, ::after, ::before {
    box-sizing: border-box;
}
*, ::after, ::before {
    box-sizing: border-box;
}
```

Snapshot:-

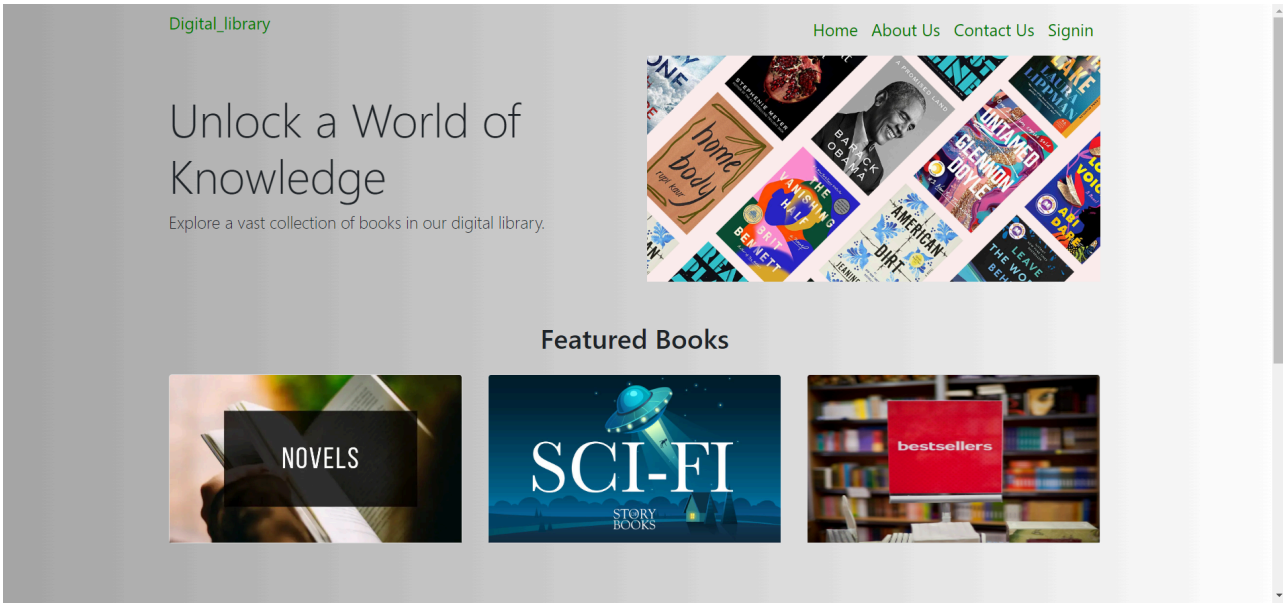


Fig 4.1: CSS design

Chapter 5

Responsive design using media queries

Media queries in CSS3 look at the capability of the device. Media queries can be used to check many things, such as:

- width and height of the viewport
- width and height of the device
- orientation (is the tablet/phone in landscape or portrait mode?)
- resolution

Using media queries are a popular technique for delivering a tailored style sheet to desktops, laptops, tablets, and mobile phones (such as iPhone and Android phones). Media queries are used for creating responsive web sites . You can also use media queries to specify that certain styles are only for printed documents or for screen readers (mediatype: print, screen, or speech).

In addition to media types, there are also media features. Media features provide more specific details to media queries, by allowing to test for a specific feature of the user agent or display device. For example, you can apply styles to only those screens that are greater, or smaller, than a certain width.

A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.

```
@media not|only mediatype and (expressions) {

    CSS-Code;

}
```

The result of the query is true if the specified media type matches the type of device the document is being displayed on and all expressions in the media query are true. When a media query is true, the corresponding style sheet or style rules are applied, following the normal cascading rules.

Unless you use the not or only operators, the media type is optional and the all type will be implied

The following example changes the background-color to light green if the viewport is 480 pixels wide or wider (if the viewport is less than 480 pixels, the background-color will be pink):

@media screen and (min-width: 480px) {

body {

background-color: lightgreen; }}

Code:-

```
@media (max-width: 768px) {
    .navbar-light .navbar-nav .nav-link {
        color: #333; /* Change text color on mobile devices */
    }
    .navbar-light .navbar-toggler-icon {
        background-color: #333; /* Change hamburger icon color on mobile devices */
    }
    .navbar-light .navbar-brand {
        font-size: 24px; /* Adjust brand logo size on mobile devices */
    }
    section {
        padding: 20px 0;
    }
     .parallax-content h1 {
        font-size: 24px;
    }
    .parallax-content p {
        font-size: 16px;
    }
}
```

Output:-

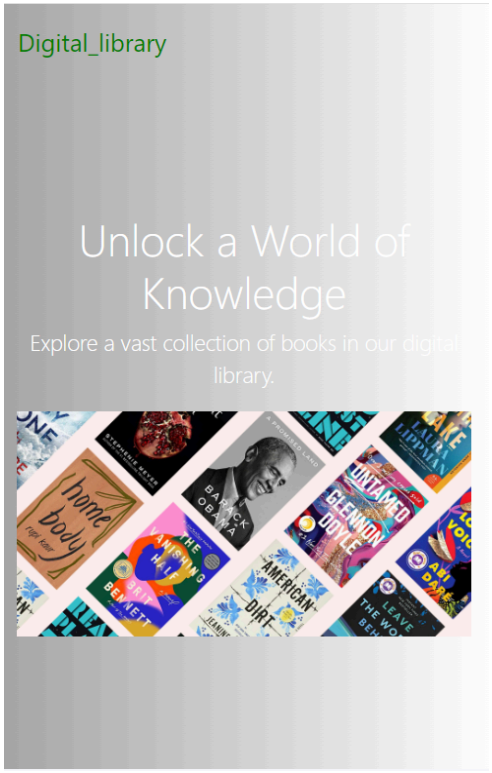


Fig 5.1: Responsive design

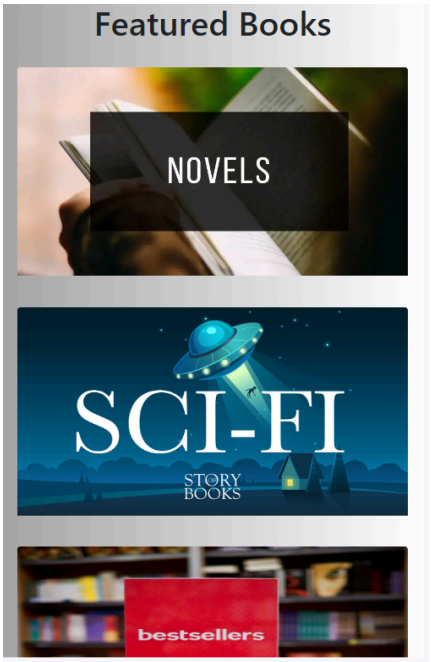


Fig 5.2: Responsive design

Chapter 6

Embedding Google Maps in web page

Google Maps is a web mapping service developed by Google. It offers satellite imagery, street maps, 360° panoramic views of streets (Street View), real-time traffic conditions (Google Traffic), and route planning for traveling by foot, car, bicycle (in beta), or public transportation. Google Maps API in June 2005[19] to allow developers to integrate Google Maps into their websites. It was a free service that didn't require an API key until June 2018 (changes went into effect on July 16), when it was announced that an API key linked to a Google Cloud account with billing enabled would be required to access the API. The API currently does not contain ads, but Google states in their terms of use that they reserve the right to display ads in the future.

By using the Google Maps API, it is possible to embed Google Maps into an external website, on to which site-specific data can be overlaid. Although initially only a JavaScript API, the Maps API was expanded to include an API for Adobe Flash applications (but this has been deprecated), a service for retrieving static map images, and web services for performing geocoding, generating driving directions, and obtaining elevation profiles. Over 1,000,000 web sites use the Google Maps API, making it the most heavily used web application development API.

The Google Maps API is free for commercial use, provided that the site on which it is being used is publicly accessible and does not charge for access, and is not generating more than 25,000 map accesses a day. Sites that do not meet these requirements can purchase the Google Maps API for Business.

The success of the Google Maps API has spawned a number of competing alternatives, including the HERE Maps API, Bing Maps Platform, Leaflet and OpenLayers via self-hosting. The Yahoo! Maps API is in the process of being shut down.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
       /* Set the size of the div element that contains the map */
      #map {
        height: 400px; /* The height is 400 pixels */
        width: 100%; /* The width is the width of the web page */
       }
    </style>
  </head>
  <body>
    <h3>My Google Maps Demo</h3>
    <!--The div element for the map -->
    <div id="map"></div>
    <script>
// Initialize and add the map
function initMap() {
  // The location of Uluru
  var uluru = {lat: -25.344, lng: 131.036};
  // The map, centered at Uluru
  var map = new google.maps.Map(
      document.getElementById('map'), {zoom: 4, center: uluru});
  // The marker, positioned at Uluru
  var marker = new google.maps.Marker({position: uluru, map: map}); }
    </script>
    <!--Load the API from the specified URL
    * The async attribute allows the browser to render the page while the API loads * The key
    parameter will contain your own API key (which is not needed for this tutorial) * The
    callback parameter executes the initMap() function
    <script async defer
    src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMap">
    </script>
  </body>
</html>
```

Code:-

```
<!DOCTYPE html>
<html>
<head>
    <title>Libraries Near Me</title>
    <style>
        /* Set the size of the map container */
        #map {
            height: 400px;
            width: 100%;
        }
        body{
        background-image: linear-gradient(90deg,#a6a6a6,#ffffff)
        }
    </style>
</head>
<body>
    <h1>Libraries Near Me</h1>
    <div id="map"></div>

    <script>
        function initMap() {
            // Create a map centered on a default location.
            var map = new google.maps.Map(document.getElementById('map'), {
                center: { lat: 40.7128, lng: -74.0060 }, // Default to New York City
                zoom: 12
            });

            // Try to get the user's location using Geolocation API
            if (navigator.geolocation) {
                navigator.geolocation.getCurrentPosition(function(position) {
                    var userLocation = {
                        lat: position.coords.latitude,
                        lng: position.coords.longitude
                    };

                    // Center the map on the user's location
                    map.setCenter(userLocation);

                    // Create a request to search for libraries nearby
                    var request = {
                        location: userLocation,
                        radius: 5000, // Search within a 5 km radius
                        type: 'library'
                    };

                    // Create a PlacesService and send the request
                    var service = new google.maps.places.PlacesService(map);
                    service.nearbySearch(request, function(results, status) {
                        if (status === google.maps.places.PlacesServiceStatus.OK) {
                            for (var i = 0; i < results.length; i++) {
                                createMarker(results[i]);
```



```
    }}};
```

```
        function createMarker(place) {
            var marker = new google.maps.Marker({
              map: map,
                position: place.geometry.location,
                title: place.name
            });

            var infowindow = new google.maps.InfoWindow({
                content: place.name
            });

            marker.addListener('click', function() {
                infowindow.open(map, marker);
            });
        }
    </script>
  <script async defer
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCgZ-LGpt1qUHVxAWrzfiC89ZQ7UdaMoiA&libraries=places&callback=initMap">
    </script>
</body>
</html>
```

Output:-

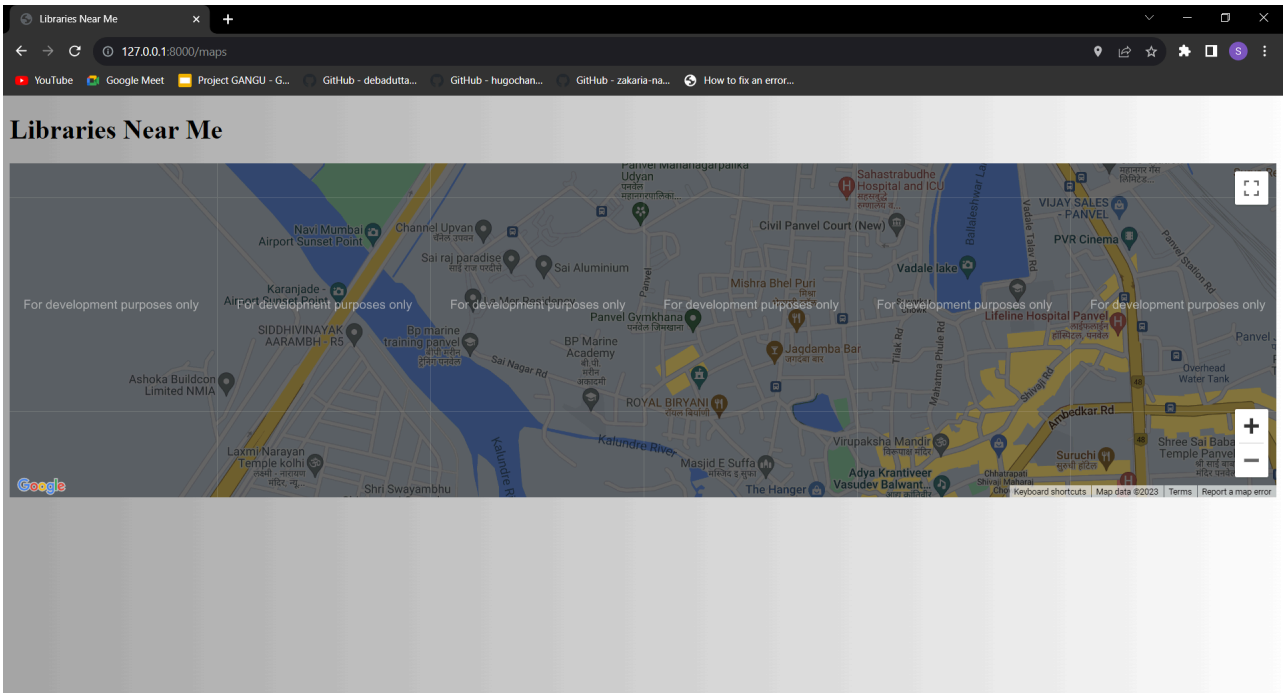


Fig 6.1: Google maps

Chapter 7

HTML5 based form validation

Forms are used in webpages for the user to enter their required details that are further send it to the server for processing. A form is also known as web form or HTML form. Form validation helps us to ensure that users fill out forms in the correct format, making sure that submitted data will work successfully with our applications.

Go to any popular site with a registration form, and you will notice that they give you feedback when you don't enter your data in the format they are expecting. You'll get messages such as:

"This field is required" (you can't leave this field blank)

"Please enter your phone number in the format xxx-xxxx" (it enforces three numbers followed by a dash, followed by four numbers)

"Please enter a valid e-mail address" (if your entry is not in the format of "somebody@example.com")

"Your password needs to be between 8 and 30 characters long, and contain one uppercase letter, one symbol, and a number"

This is called form validation — when you enter data, the web application checks it to see that the data is correct. If correct, the application allows the data to be submitted to the server and (usually) saved in a database; if not, it gives you an error message explaining what corrections need to be made. Form validation can be implemented in a number of different ways.

We want to make filling out web forms as easy as possible. So why do we insist on validating our forms? There are three main reasons:

We want to get the right data, in the right format — our applications won't work properly if our user's data is stored in the incorrect format, or if they don't enter the correct information, or omit information altogether.

We want to protect our users' accounts — by forcing our users to enter secure passwords, it makes it easier to protect their account information.

We want to protect ourselves — there are many ways that malicious users can misuse unprotected forms to damage the application they are part of (see Website security).

Different types of form validation

There are two different types of form validation which you'll encounter on the web:

Client-side validation is validation that occurs in the browser before the data has been submitted to the server. This is more user-friendly than server-side validation as it gives an instant response. This can be further subdivided:

JavaScript validation is coded using JavaScript. It is completely customizable.

Built-in form validation using HTML5 form validation features. This generally does not require JavaScript. Built-in form validation has better performance, but it is not as customizable as JavaScript.

Server-side validation is validation which occurs on the server after the data has been submitted. Server-side code is used to validate the data before it is saved into the database. If the data fails authentication, a response is sent back to the client to tell the user what corrections to make. Server-side validation is not as user-friendly as client-side validation, as it does not provide errors until the entire form has been submitted. However, server-side validation is your application's last line of defence against incorrect or even malicious data. All popular server-side frameworks have features for validating and sanitizing data (making it safe).

1. Specialized Input Types

HTML5 introduced several new input types. They can be used to create input boxes, which will accept only a specified kind of data.

The new input types are as follows:
Color, date, datetime, email , month .number, range . search, tel , time , url , week To

use one of the new types, include them as the value of the type attribute: <input

type="email"/>

2. Required Fields

By simply adding the "required" attribute to a <input>, <select> or <textarea>, you tell the browser that a value must be provided in this field. Think of this as the red asterisk* we see in most registration forms.

<input type="checkbox" name="terms" required >

3. Limits

We can set some basic limitations like max length and minimum and maximum values for number fields. To limit the length of input fields and textareas, use the "maxlength" attribute. What this does is to forbid any string longer than the field's "maxlength" value to be entered at all. If you try and paste a string witch exceeds this limit, the form will simply clip it.

<input type="text" name="name" required maxlength="15">

The <input type="number"> fields use "max" and "min" attributes to create a range of possible values - in our example we've made the minimum allowed age to be 18 (too bad you can be whatever age you want on the internet).

<input type="number" name="age" min="18" required>

Code:-

```
<div class="container">
    <div class="row">
        <div class="col-md-6 offset-md-3 login-container">
            <h2 class="login-heading">Log In to e-Library</h2>
            <form method="POST">
                {% csrf_token %}
                <div class="form-group">
                    <label for="email">Email</label>
                    <input type="email" class="form-control" id="email" name="email"
placeholder="Enter Email" Required>
                </div>
                <div class="form-group">
                    <label for="pass1">Password</label>
                    <input type="password" class="form-control" id="pass1"
name="pass1" placeholder="Enter Your Password" Required>
                </div>
                <button type="submit" class="btn btn-primary btn-lg login-button">Log
In</button>
            </form>
```

```
 <p class="create-account-link">Don't have an account? <a href="/registration">Create
one</a></p>
        </div>
    </div>
</div>
```

Output:-

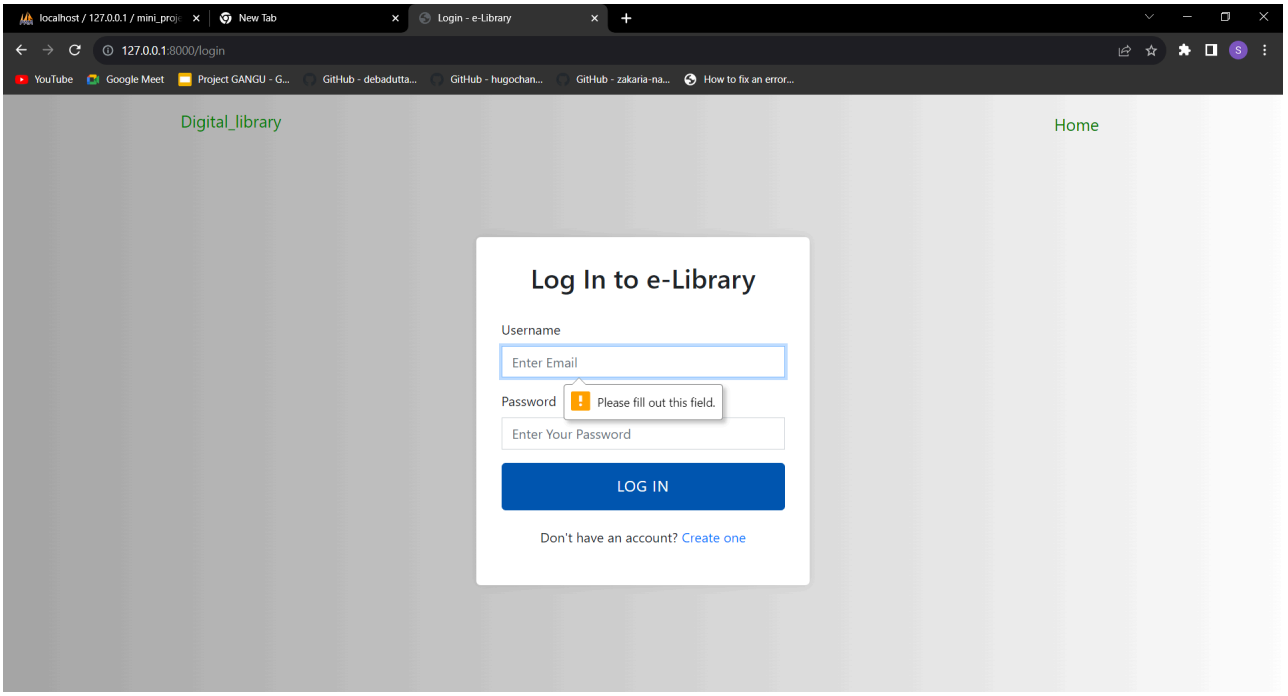


Fig 7.1: HTML based form validation

Chapter 8

Javascript based form validation

Validating form input with JavaScript is easy to do and can save a lot of unnecessary calls to the server as all processing is handled by the web browser. It can prevent people from leaving fields blank, from entering too little or too much or from using invalid characters.

Forms validation on the client-side is essential — it saves time and bandwidth, and gives you more options to point out to the user where they've gone wrong in filling out the form. Having said that, I don't mean that you don't need server-side validation. People who visit your site may use an old browser or have JavaScript disabled, which will break client-only validation. Client and server-side validation complement each other, and as such, they really shouldn't be used independently.

Why is Client Side Validation Good?

There are two good reasons to use client-side validation:

1. It's a fast form of validation: if something's wrong, the alarm is triggered upon submission of the form.
2. You can safely display only one error at a time and focus on the wrong field, to help ensure that the user correctly fills in all the details you need.

Two Major Validation Approaches

1. Display the errors one by one, focusing on the offending field
2. Display all errors simultaneously, server-side validation style

While displaying all errors simultaneously is required for server-side validation, the better method for validation on the client-side is to show one error at a time. This makes it possible to highlight only the field that has been incorrectly completed, which in turn makes revising and successfully submitting the form much easier for the visitor. If you present users with all errors at the same time, most people will try to remember and correct them at once, instead of attempting to re-submit after each correction.

```
function validateForm() {

  var x = document.forms["myForm"]["fname"].value;
```

```
    if (x == "") {
        alert("Name must be filled out");
        return false;
    }
}
<form name="myForm" action="/action_page.php" onsubmit="return validateForm()"
method="post">
Name: <input type="text" name="fname">
<input type="submit" value="Submit">
</form>
function checkpassword(pform1){
var str=pform1.password.value;
//check required fields
//password should be minimum 4 chars but not greater than 8
if ((str.length < 4) || (str.length > 8)) {
function checkpassword(pform1){
var str=pform1.password.value;

//check required fields

//password should be minimum 4 chars but not greater than 8

if ((str.length < 4) || (str.length > 8)) {
alert("Invalid password length.")
pform1.password.focus()
return false
```

```
}
}
function checkemailphone(pform1){
var email = pform1.email.value;
var phone = pform1.phone.value;
var cleanstr = phone.replace(/[().- ]/g, '');
var validemail =/^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+.[a-zA-Z]{2,4}$/;
if(!(validemail.test(email))){
alert("Invalid email address")
pform1.email.focus()
return false
}
//check phone number
if (isNaN(parseInt(cleanstr))) {
alert("The phone number contains unwanted characters.") }
}
```

Code:-

```
<script>
    function validateForm() {
  let x = document.forms["myForm"]["fname"].value;
  if (x == "") {
    alert("Name must be filled out");
    return false;
  }
}
</script>
```

Output:-

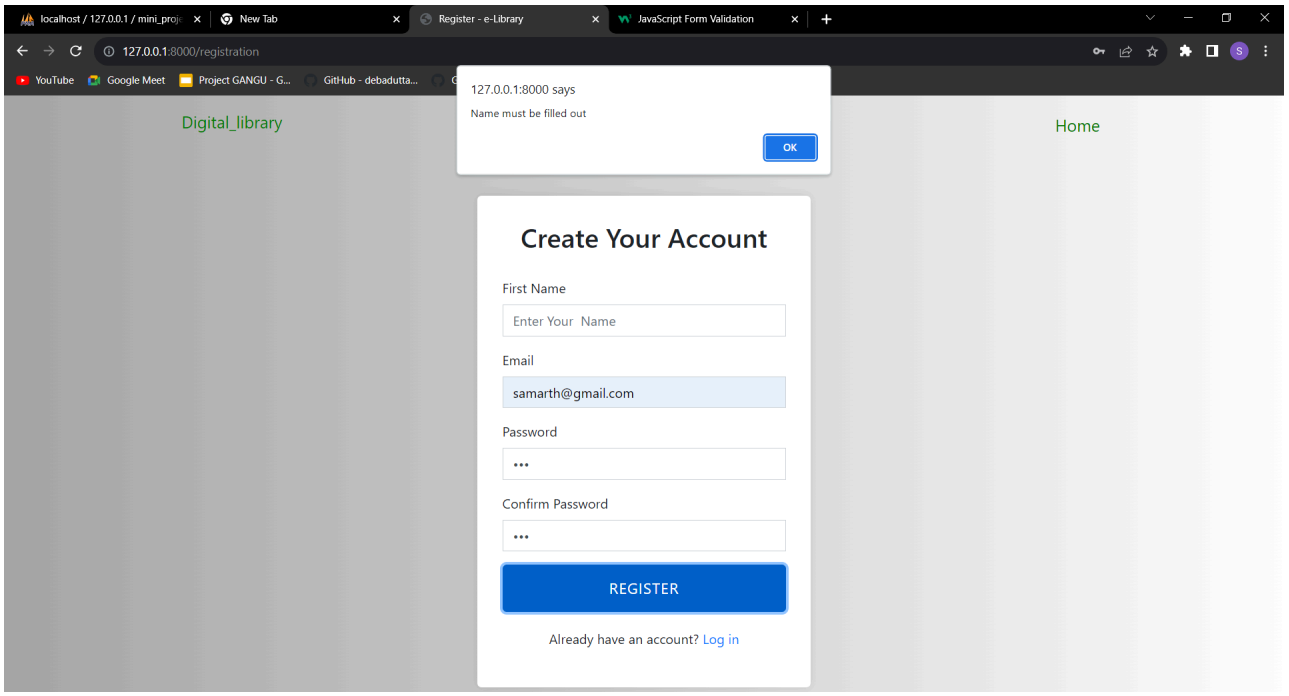


Fig 8.1: Javascript based form validation

Chapter 9

Server side programming using PHP

PHP stands for Hypertext Preprocessor. PHP is a very popular and widely-used open source server-side scripting language to write dynamically generated web pages. PHP was originally created by Rasmus Lerdorf in 1994. It was initially known as Personal Home Page.

PHP scripts are executed on the server and the result is sent to the web browser as plain HTML. PHP can be integrated with the number of popular databases, including MySQL, PostgreSQL, Oracle, Microsoft SQL Server, Sybase, and so on.

What You Can Do with PHP

- There are lot more things you can do with PHP.
- You can generate pages and files dynamically.
- You can create, open, read, write and close files on the server.
- You can collect data from a web form such as user information, email, phone no, etc. ● You can send emails to the users of your website.
- You can send and receive cookies to track the visitor of your website.
- You can store, delete, and modify information in your database.
- You can restrict unauthorized access to your website.
- You can encrypt data for safe transmission over internet.
- The list does not end here, there are many other interesting things that you can do with PHP. You will learn about all of them in detail in upcoming chapters.

Advantages of PHP over Other Languages

- If you're familiar with other server-side languages like ASP.NET or Java, you might be wondering what makes PHP so special. There are several advantages why one should choose PHP.
- Easy to learn: PHP is easy to learn and use. For beginner programmers who just started out in web development, PHP is often considered as the preferable choice of language to learn. ● Open source: PHP is an open-source project. It is developed and maintained by a worldwide community of developers who make its source code freely available to download and use.

- Portability: PHP runs on various platforms such as Microsoft Windows, Linux, Mac OS, etc. and it is compatible with almost all servers used today such Apache, IIS, etc. ● Fast Performance: Scripts written in PHP usually execute or runs faster than those written in other scripting languages like ASP, Ruby, Python, Java, etc.
- Vast Community: Since PHP is supported by the worldwide community, finding help or documentation related to PHP online is extremely easy.

Setting Up a Local Web Server

PHP script execute on a web server running PHP. So before you start writing any PHP program you need the following program installed on your computer.

1. The Apache Web server
2. The PHP engine
3. The MySQL database server

You can either install them individually or choose a pre-configured package for your operating system like Linux and Windows. Popular pre-configured package are XAMPP and WampServer.Creating Your First PHP Script

```
<!DOCTYPE HTML>
<html>
<head>
   <title>PHP Application</title>
</head>
<body>
<?php
// Display greeting message
echo 'Hello World!';
?>
</body>
</html>
```

Code:-

```
<?php
$servername = "localhost";
$username = "sam3";
$password = "samarth";
$dbname = "ip2";
$name = $_REQUEST['name'];
$email =  $_REQUEST['email'];
$pass1 = $_REQUEST['pass1'];
```

```
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$name = $_REQUEST['name'];
$email =  $_REQUEST['email'];
$pass1 = $_REQUEST['pass1'];
$sql = "INSERT INTO user2 (Name,Email,Password)
VALUES ('$name', '$email', '$password')";

if ($conn->query($sql) === TRUE) {
  echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
<html>
<head><title>Registration</title></head>
<body>
<div class="container">
    <div class="row">
        <div class="col-md-6 offset-md-3 registration-container">
            <h2 class="registration-heading">Create Your Account</h2>
            <form method="POST" onsubmit="return validateForm()" name="myForm">
                <div class="form-group">
                    <label for="fname">First Name</label>
                    <input type="text" class="form-control" id="fname" name="fname"
placeholder="Enter Your  Name" >
             <div class="form-group">
                    <label for="email">Email</label>
                    <input type="email" class="form-control" id="email" name="email"
placeholder="Enter Your Email Address" Required>
                </div>
                <div class="form-group">
                    <label for="pass1">Password</label>
                    <input type="password" class="form-control" id="pass1"
name="pass1" placeholder="Create Your Password" Required>
                </div>
                <div class="form-group">
                    <label for="pass2">Confirm Password</label>
                    <input type="password" class="form-control" id="pass2"
name="pass2" placeholder="Confirm Your Password" Required>
                </div>
                <button type="submit" class="btn btn-primary btn-lg
registration-button">Register</button>
            </form>
            <p class="login-link">Already have an account? <a href="/login">Log
in</a></p>
        </div>
```

```
    </div>
</div>
</body>
</html>
```

Output:-

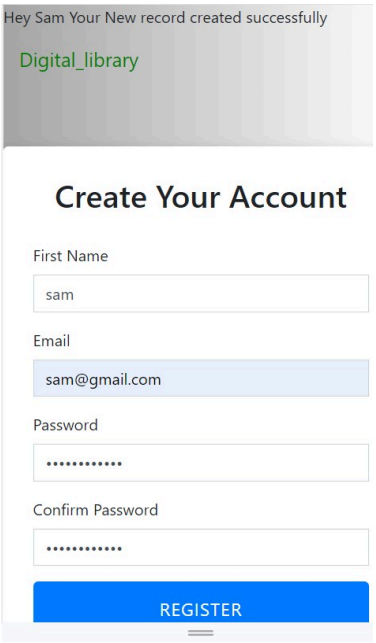


Fig 9.1: server side programming

Chapter 10

PHP MySQL database operations

With PHP, you can connect to and manipulate databases MySQL is the most popular database system used with PHP.

What is MySQL?

- MySQL is a database system used on the web
- MySQL is a database system that runs on a server
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, and easy to use
- MySQL uses standard SQL
- MySQL compiles on a number of platforms
- MySQL is free to download and use

The data in a MySQL database are stored in tables. A table is a collection of related data, and it consists of columns and rows. Databases are useful for storing information categorically. A company may have a database with the following tables:

- Employees
- Products
- Customers
- Orders

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
// Create connection
$conn = mysqli_connect($servername, $username, $password);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

If you want to use PHP to query your MySQL database you can do that by either entering the MySQL query command in the PHP script or define the command as a variable and use the variable when needed

mysqli_query($query);

The command can be repeated again in the source code. All you need to do is to change the $query variable.

For example, here is the complete code that could be used to create a MySQL table in PHP:

```
<?php
$username = "your_username";
$password = "your_password";
$database = "your_database";
$mysqli = new mysqli("localhost", $username, $password, $database);
$query="CREATE TABLE tablename(id int(6) NOT NULL auto_increment,first varchar(15) NOT NULL,last varchar(15) NOT NULL,field1-name varchar(20) NOT NULL,field2-name varchar(20)NOT NULL,field3-name varchar(20) NOT NULL,field4-name varchar(30) NOT NULL, field5-name varchar(30)NOT NULL,PRIMARY KEY (id),UNIQUE id (id),KEY id_2 (id))";
$mysqli->query("$query");
$mysqli->close();
?>
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
    // output data of each row
```

```
    while($row = mysqli_fetch_assoc($result)) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>"; }
} else {
echo "0 results";
}
mysqli_close($conn);
?>
```

Code:-

```
<?php
if(isset($_POST['fname'])){
    $server = "localhost";
    $username = "demo";
    $password = "demo";
    $con = mysqli_connect($server, $username, $password);

    $name = $_POST['fname'];
    $email = $_POST['email'];
    $pass1 = $_POST['pass1'];

          $sql  =  "INSERT  INTO  `ip2`.`user`  (`Name`,`Email`,`Password`)  VALUES
('$name','$email','$pass1');";

    //echo $sql;
    if($con->query($sql) == TRUE){
        //echo "success";
    }
    else{
        echo "error";
    }
    $con->close();
}
?>
```

Output:-

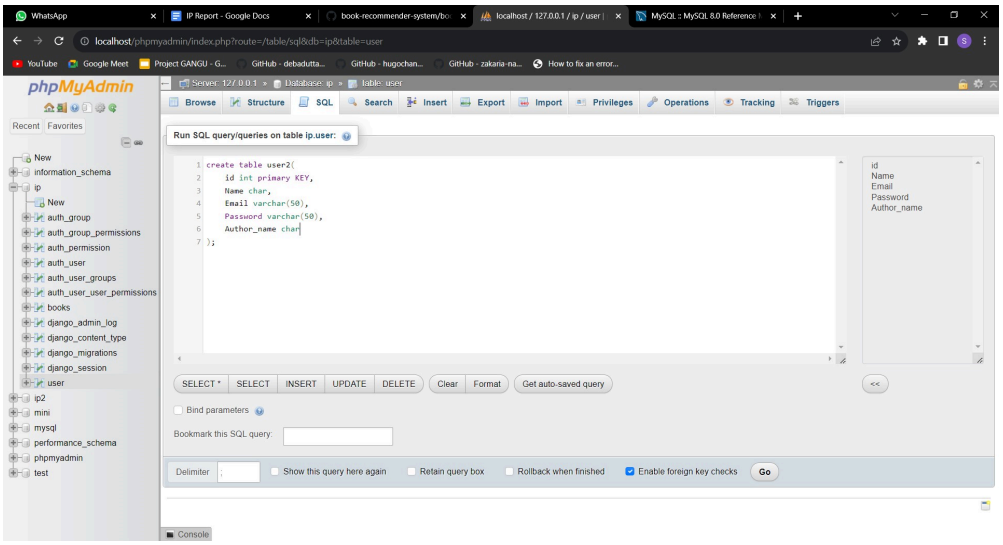


Fig 10.1: Creation of table in database

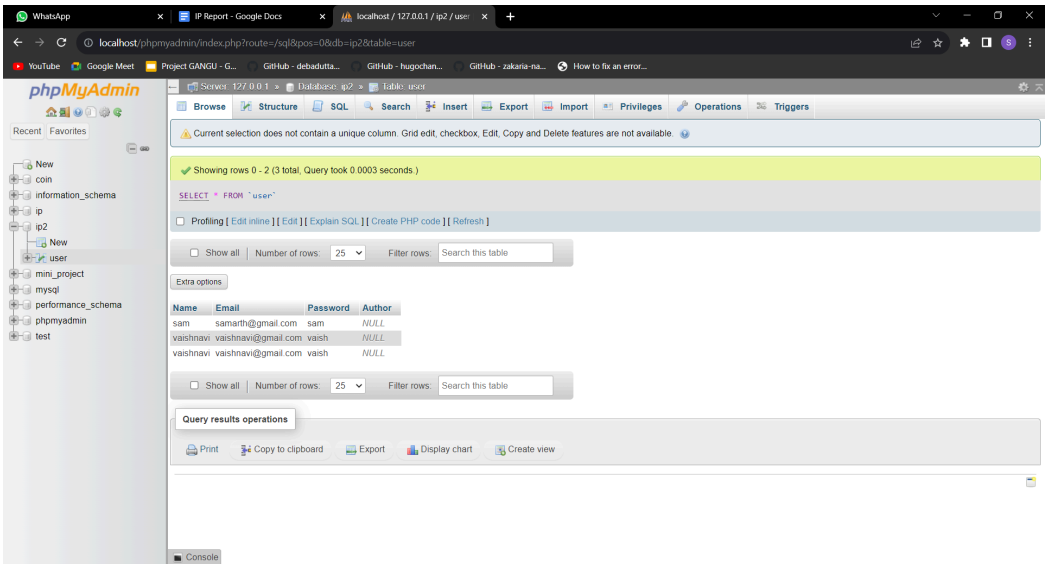


Fig 10.2: Database

Chapter 11

Web hosting

Web hosting is a service that allows organizations and individuals to post a website or web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain into their browser. Their computer will then connect to your server and your web pages will be delivered to them through the browser.

Most hosting companies require that you own your domain in order to host with them. If you do not have a domain, the hosting companies will help you purchase one.

Here are some features you should be expecting from your hosting provider:

Email Accounts : As mentioned earlier, most hosting providers require users to have their own domain name. With a domain name (e.g. www.yourwebsite.com) and email account features provided by your hosting company, you can create domain email accounts (e.g. yourname@yourwebsite.com).

FTP Access : The use of FTP lets you upload files from your local computer to your web server. If you build your website using your own HTML files, you can transfer the files from your computer to the web server through FTP, allowing your website to be accessed through the internet.

https://elib-0cca9de97c28.herokuapp.com/

Snapshot of Landing page

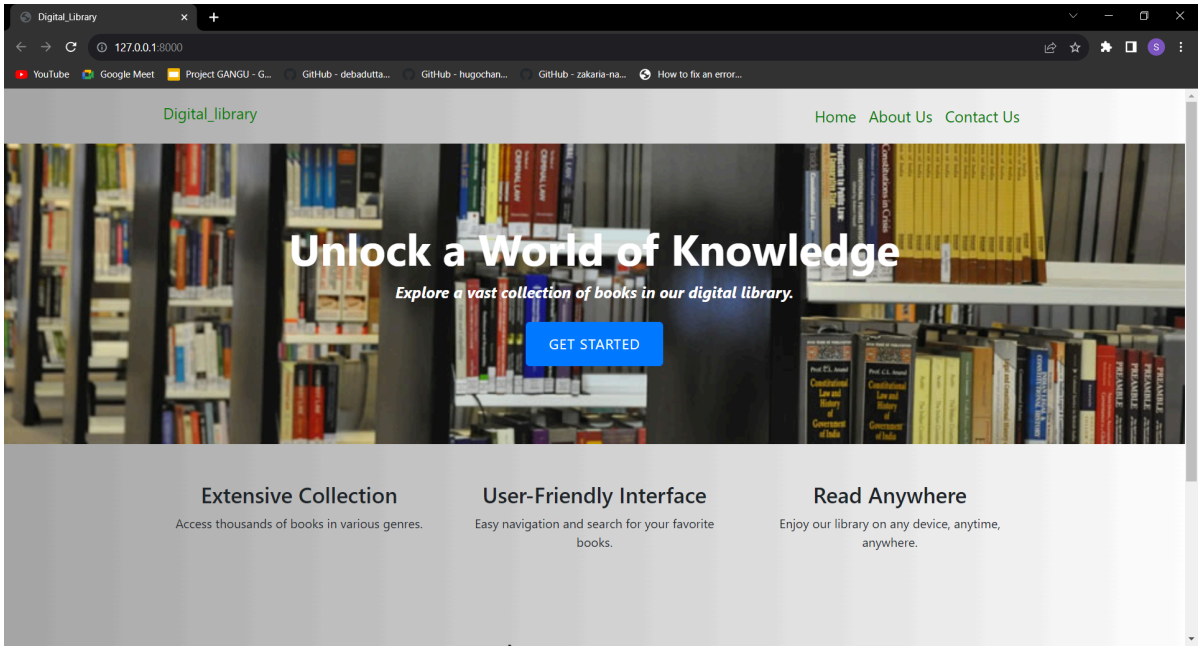


Fig 11: Landing Page

Snapshot of Login Page

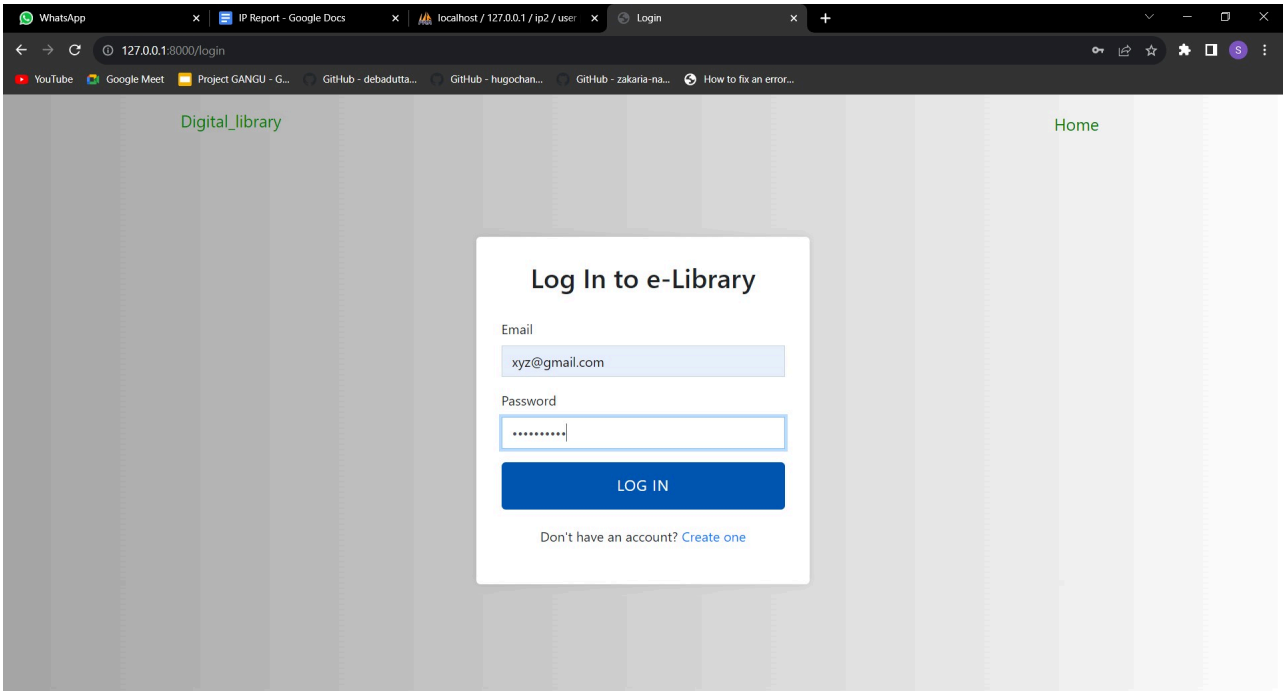
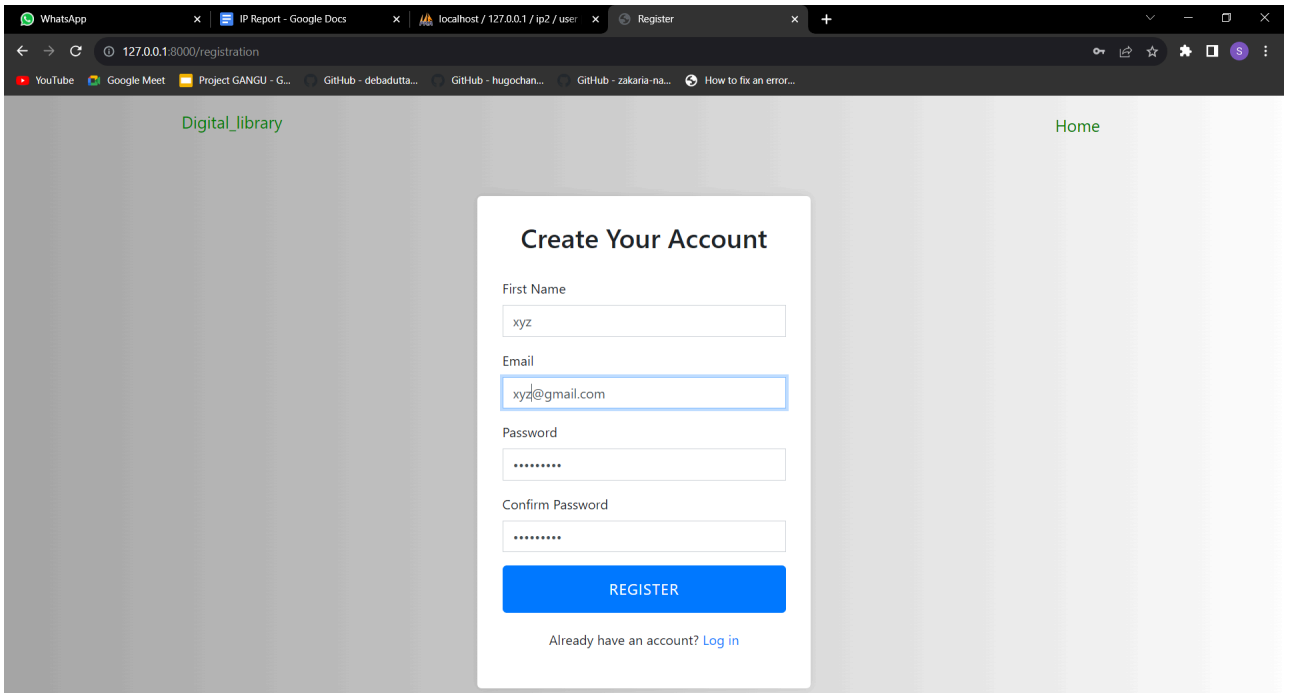


Fig 12: Login page

Snapshot of Registration Page



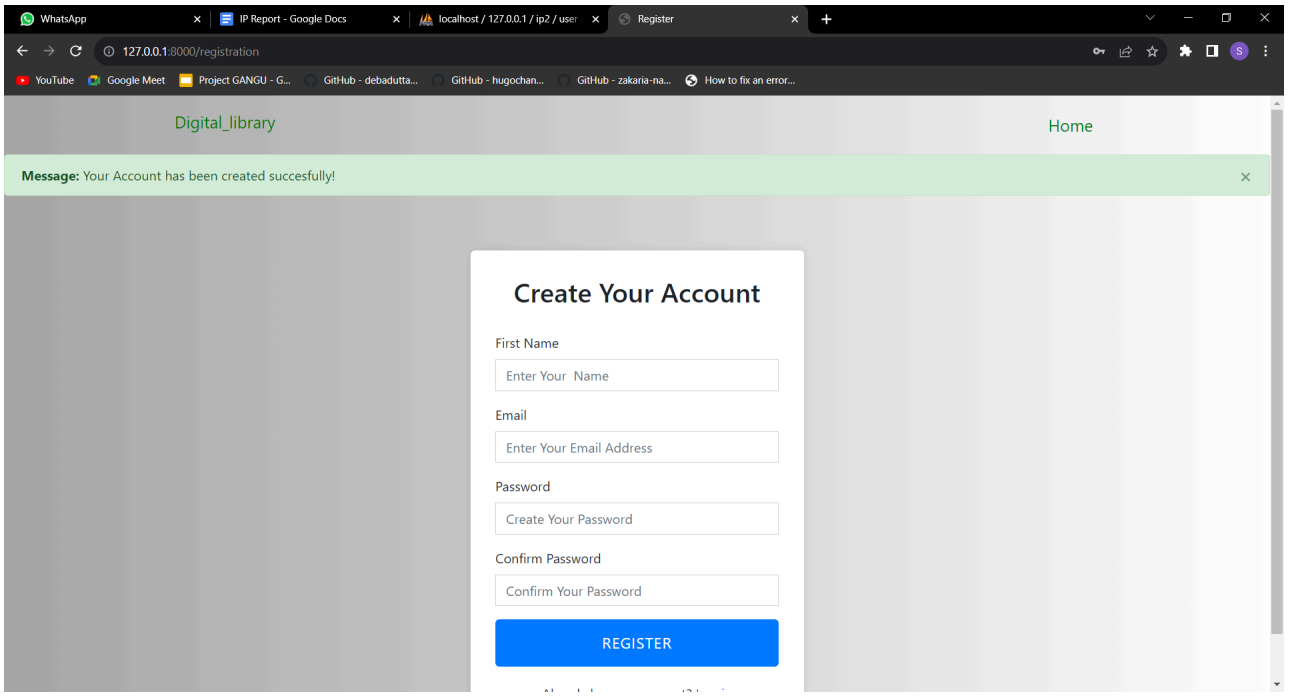


Fig 13: Registration page

Snapshot of About us page

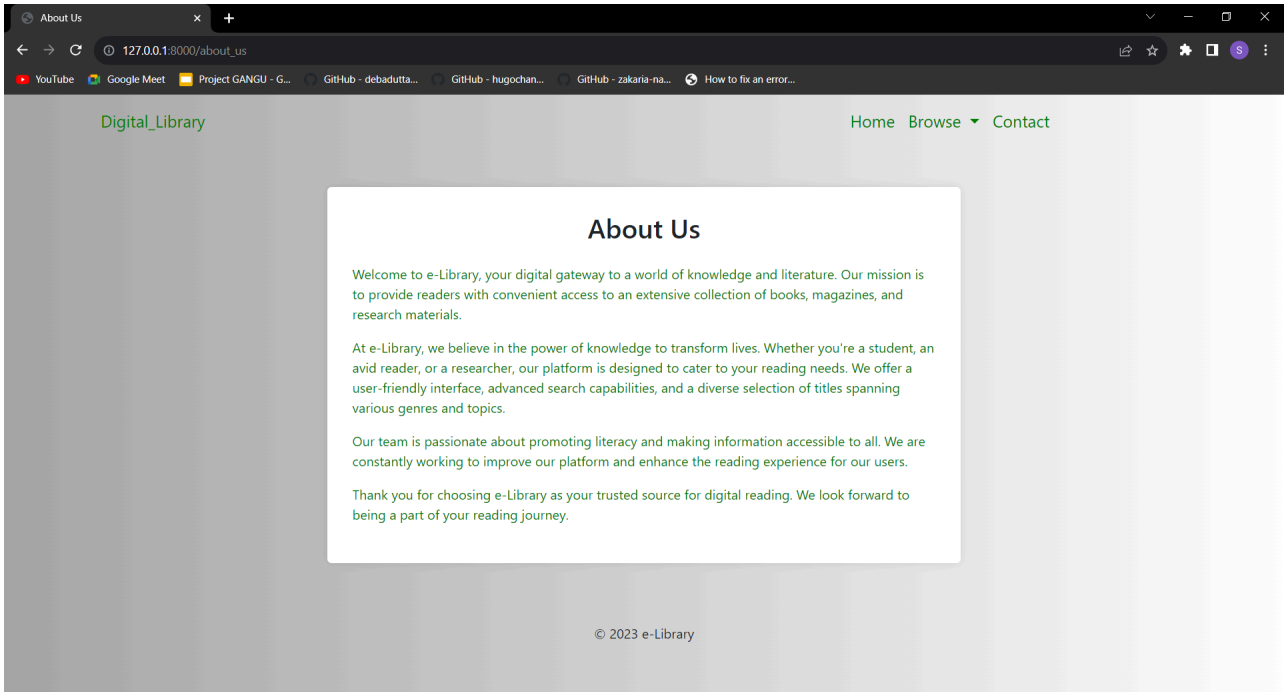


Fig 14: About us

Snapshot of Contact page

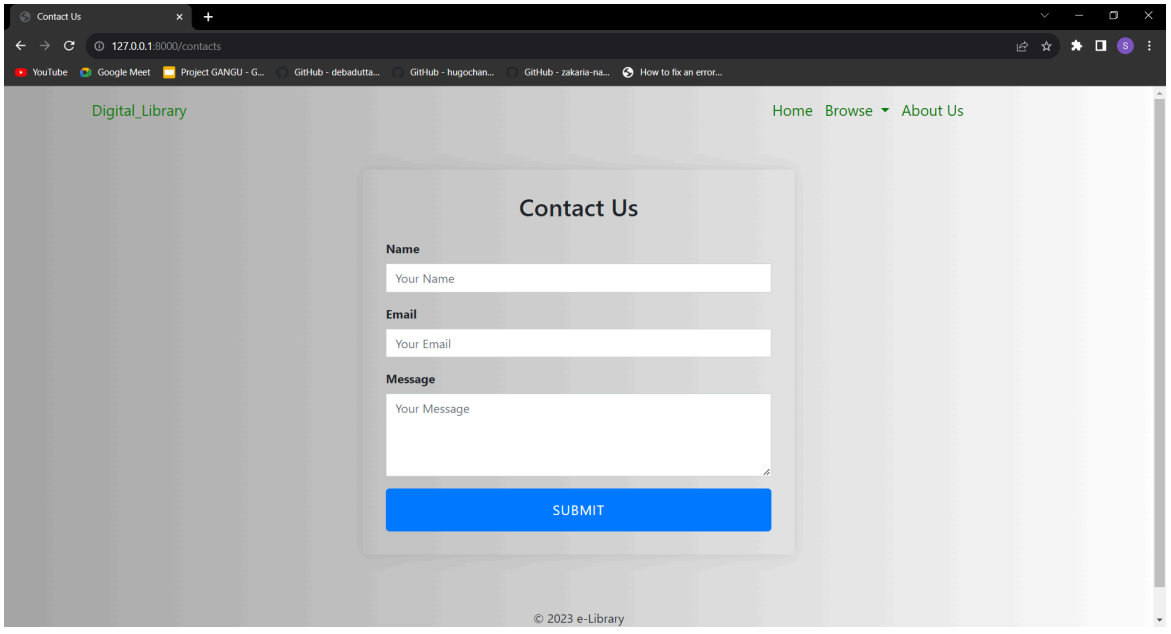


Fig 15: Contact us

Snapshot of User Home page

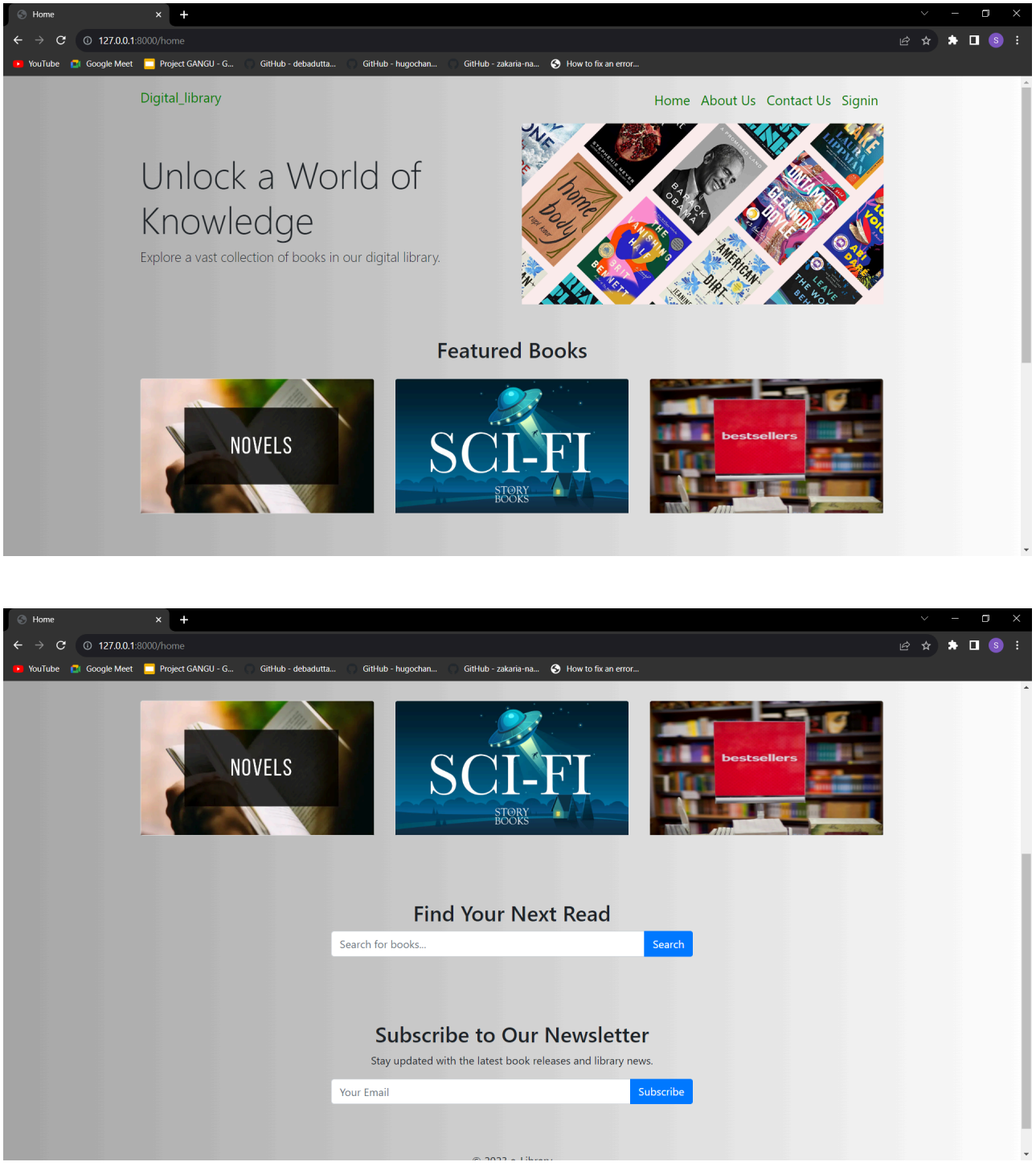


Fig 16: Home page

Acknowledgement

We would like to express our special thanks to **Prof. Mimi Cherian**, our Internet Programming Mini project guide who guided us through the project and who helped us in applying the knowledge that we have acquired during the semester and learning new concepts.

We would like to express our special thanks to **Dr. Satishkumar Varma,** Head, Department of Information Technology, who gave us the opportunity to do this Internet Programming Mini project because of which we learned new concepts and their application.

Finally we would like to express our special thanks to Principal **Dr. Sandeep Joshi** who gave us the opportunity and facilities to conduct this Internet Programming Mini project.

Shravni Patil
Samarth Patil
Sameer Shahi