

**Mini-project Report on**  
**“Travel Management System”**

**Submitted by:**

Dharmik Tank  
Rahul Metre  
Ananya Sharma  
Piyush Motwani

Roll Number: PC01  
Roll Number: PC06  
Roll Number: PC09  
Roll Number: PC13

**Under the Guidance**

**of Faculty Name**

**Prof. Hanmat Magar**

**At**



Dr. Vishwanath Karad

**MIT WORLD PEACE  
UNIVERSITY | PUNE**

TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

**School of Computer Engineering and Technology**

## **Acknowledgements**

We would like to express our deep gratitude and sincere thanks to our DBMS Professor **Prof. Hanmat Magar** for their constant valuable advice and constructive inputs which enhanced the quality of our work. This project could not have been completed successfully without their help.

Lastly, we would also like to thank our parents for their love and guidance without which, nothing can be accomplished.

## **Abstract Idea**

A Travel Management System is a web-based application designed to streamline and automate various travel-related processes. Using HTML, CSS, PHP, and MySQL, a Travel Management System can provide a comprehensive platform for travel agencies or companies to manage their operations efficiently.

The system would allow users to search and book flights according to their convenience, providing them with real-time availability and pricing information. It would also facilitate the management of travel itineraries, including scheduling, reservations, and ticketing. Users can access their travel details, view invoices, and make payments securely through the system.

Furthermore, the system can incorporate features like expense tracking, reporting, and analytics to assist in budgeting and cost analysis. It can integrate with external APIs for fetching data from third-party providers, ensuring access to the latest information.

With HTML and CSS, the system's user interface can be developed to offer an intuitive and visually appealing experience. PHP would handle the server-side scripting, enabling dynamic functionality and database interactions using MySQL for data storage and retrieval.

In summary, a Travel Management System using HTML, CSS, PHP, and MySQL would provide a robust and user-friendly platform for efficient travel planning, booking, and management.

## INDEX

S. No.	Content	Page No.
1.	INTRODUCTION	5
2.	PROBLEM DEFINITION	7
3.	System Flow Architecture with Database Design (ER diagram)	9
4.	GUI (Screen Shots) with client side validations	12
5.	Server-side database handling details	17
6.	Conclusions	21
7.	. APPENDIX	23

## INTRODUCTION

The travel industry is highly dynamic and demands efficient management of various travel-related processes. To address this need, a Travel Management System (TMS) powered by HTML, CSS, PHP, and MySQL can revolutionize how travel agencies and companies operate. This report provides an overview of the benefits and functionalities of a TMS, highlighting its potential to streamline travel operations.

The purpose of a TMS is to automate and centralize travel management tasks, enabling users to search, book, and manage flights. By leveraging HTML and CSS, the system can offer a visually appealing and user-friendly interface. PHP serves as the backbone, facilitating server-side scripting and dynamic functionality, while MySQL handles data storage and retrieval.

### **Key Features:**

- 1. Real-time Availability:** Users can access up-to-date information on flight schedules, hotel availability, and rental car options, ensuring accurate bookings.
- 2. Itinerary Management:** The TMS allows users to create and manage travel itineraries, including scheduling, reservations, and ticketing, all in one place.
- 3. Secure Payments:** The system provides a secure payment gateway, allowing users to make payments for bookings and view invoices conveniently.
- 4. Expense Tracking:** Integrated expense tracking features enable users to monitor travel expenses and generate comprehensive reports for budgeting and cost analysis.

This application is developed to provide best traveling services to the customers and travel agents. We have developed tours and travel management systems to provide a search platform where a tourist can find their tour places according to their choices. This system also helps to promote responsible and interesting tourism so that people can enjoy their holidays at their favorable places. This system also helps to develop tourism with different cultures so that they enrich the tourism experience and build pride. We develop this system to create and promote forms of tourism that provide healthy interaction opportunities for tourists and locals and increase better understanding of different cultures, customs, lifestyles, traditional knowledge and beliefs. This system also provides a better way to connect with various events.

## PROBLEM DEFINITION

The current travel management process lacks an integrated system for booking train, flight, and bus tickets, resulting in inefficiencies, confusion, and inconvenience for travelers. The absence of a centralized platform specifically designed for these ticket bookings poses the following problems:

**1. Fragmented Booking Process:** Travelers are required to visit multiple websites or physical ticket counters to book train, flight, and bus tickets separately. This fragmented process leads to time wastage, increases the chances of errors, and lacks a unified view of the travel options.

**2. Limited Real-time Information:** The availability, schedules, and prices of train, flight, and bus tickets are scattered across various sources, making it challenging to access up-to-date information. This lack of real-time data can result in missed opportunities, incorrect bookings, and dissatisfaction among travelers.

**3. Inefficient Comparison and Selection:** Without a centralized system, travelers face difficulties in comparing and selecting the most suitable options for their journey. This leads to confusion, longer decision-making processes, and the possibility of not making informed choices.

**4. Payment and Ticket Management:** Managing payments and tracking ticket details across different platforms becomes cumbersome for both travelers and travel agencies. It increases the chances of payment errors, miscommunication, and difficulties in handling cancellations or modifications.

**5. Lack of Personalization and Customization:** Existing systems often lack personalized features such as preferred seat selection, special requests, and tailored travel itineraries. This hampers the ability to provide a customized experience for travelers, resulting in reduced satisfaction.

To address these challenges, the development of a comprehensive Travel Management System with integrated train, flight, and bus ticket bookings is crucial. Such a system would streamline the ticket booking process, provide real-time information, facilitate comparison and selection, simplify payment and ticket management, and enhance the overall travel experience for users.



## System Flow Architecture with Database Design (ER diagram)

### Entities:

#### 1. NewBooking:

- Attributes: Alert

#### 2. Train:

- Attributes: TrainNumber (Primary Key), TrainName, Source\_City, Destination\_City, Cost

#### 3. Flight:

- Attributes: FlightNumber (Primary Key), FlightName, Source\_City, Destination\_City, Cost

#### 4. Bus:

- Attributes: BusNumber (Primary Key), BusName, Source\_City, Destination\_City, Cost

#### 5. Booking Confirmation(Flight):

- Attributes: Confirmation Status, Total Cost

#### 6 Booking Confirmation(Train):

- Attributes: Confirmation Status, Total Cost

#### 7.. Booking Confirmation(Bus):

- Attributes: Confirmation Status, Total Cost

## **Relationships:**

### **1. User-Train (One-to-Many):**

- Each user can book multiple train tickets.
- Foreign Key: User\_ID (in User entity)

### **2. User-Flight (One-to-Many):**

- Each user can book multiple flight tickets.
- Foreign Key: User\_ID (in User entity)

### **3. User-Bus (One-to-Many):**

- Each user can book multiple bus tickets.
- Foreign Key: User\_ID (in User entity)

### **4. Train-Ticket (One-to-Many):**

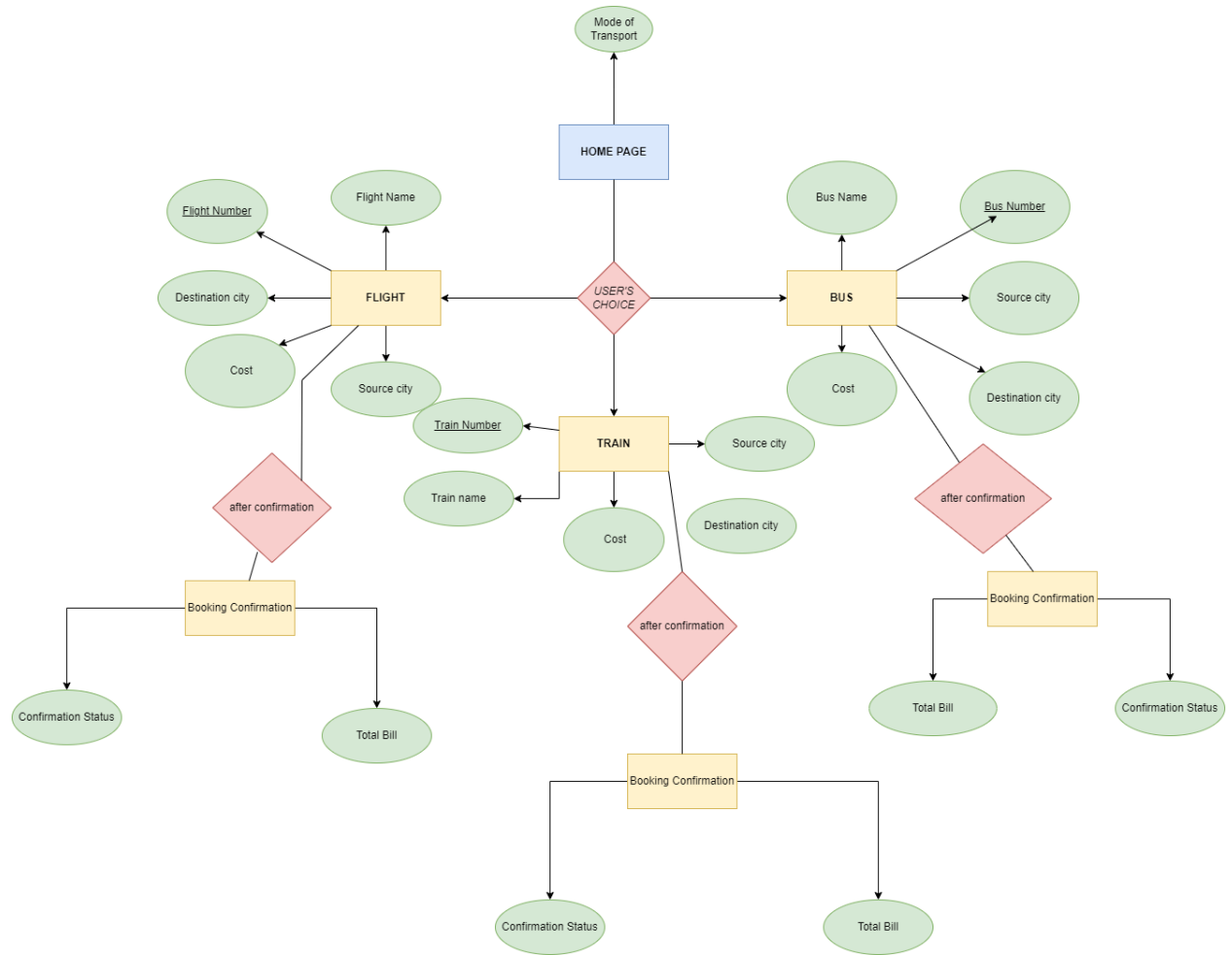
- Each train can have multiple tickets.
- Foreign Key: Train\_ID (in Train entity)

### **5. Flight-Ticket (One-to-Many):**

- Each flight can have multiple tickets.
- Foreign Key: Flight\_ID (in Flight entity)

### **6. Bus-Ticket (One-to-Many):**

- Each bus can have multiple tickets.
- Foreign Key: Bus\_ID (in Bus entity)



### **Entity-Relationship Diagram Link:**

<https://drive.google.com/file/d/17JvnFrIv9ZvQXBk6Kyn1nG1XWV7LFvgM/view?usp>

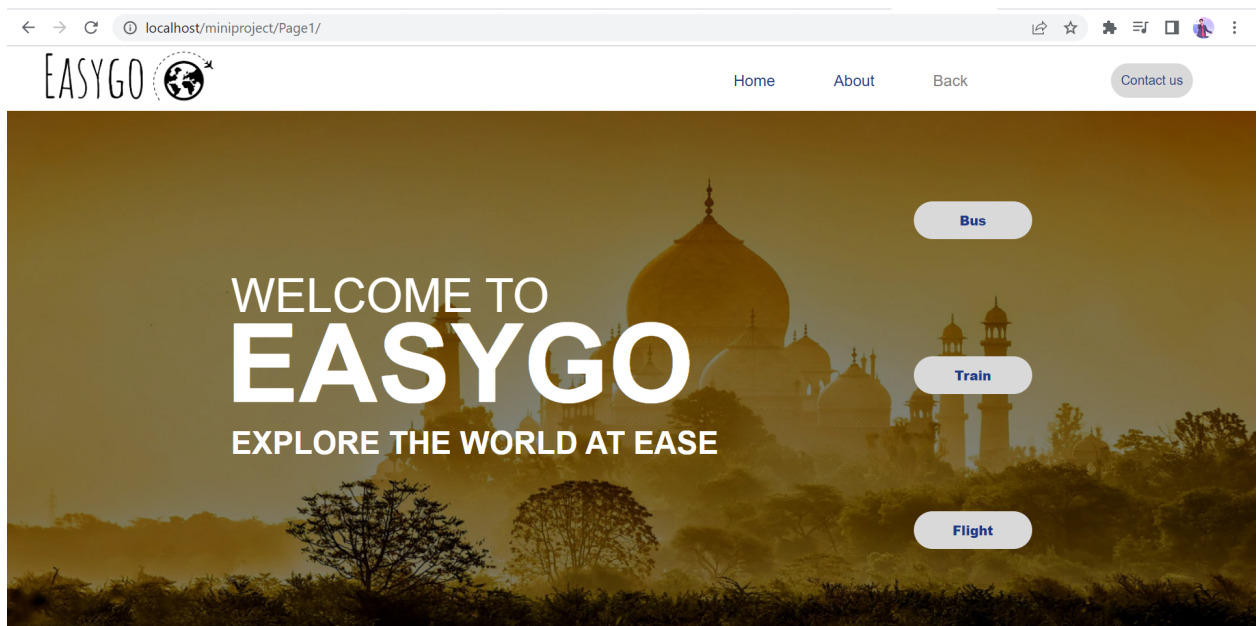
[=sharing](#)

## GUI (Screen Shots) with client side validations

In a travel management system project, client-side validations can be implemented using JavaScript to ensure that user input on the client side meets the specified criteria before submitting the data to the server. Here are some examples of client-side validations for the ticket booking process:

### 1. Type of Travel Validation:

- Make sure that the user is selecting either Bus / Train / Flight travel



### 2. Date and Time Validation:

- Validate that the departure and arrival dates are appropriate.
- Check that the number of seats are available to proceed with booking.

localhost/miniproject/Page4/

Home About Back Contact us

## Where would you like to go ?

**From**

Select Your Source

**To**

Select Your Destination

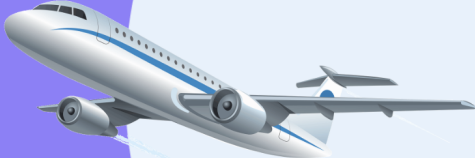
**Date**

dd-mm-yyyy

**Number of Passengers**

1(min)-10(max)

Submit



localhost/miniproject/Page6/

Home About Back Contact us

## Where would you like to go ?

**From**

Select Your Source

**To**

Select Your Destination


**Date**

dd-mm-yyyy

**Number of Passengers**

1(min)-10(max)

Submit



← → ↻ localhost/miniproject/Page5/ ☆ ⚙️ ☰ 🖨️ 👤 ⋮

Home About Back [Contact us](#)

Where would you like to go ?


From  
Select Your Source ▼

To  
Select Your Destination ▼

Date  
dd-mm-yyyy 📅

Number of Passengers  
1(min)-10(max)

Submit



### 3. Fare Validation:

- Validate that the fare is a positive number and within a specific range.

**Here are the available travel options based on your choices**

Bus No.	Bus Name	Source	Destination	Cost	Confirm
2109	Shivneri	Mumbai	Bangalore	700	<a href="#">Book</a>
2110	Neeta	Mumbai	Bangalore	750	<a href="#">Book</a>
2111	Redbus	Mumbai	Bangalore	800	<a href="#">Book</a>

**Here are the available travel options based on your choices**

Flight No.	Flight Name	Source	Destination	Cost	Confirm
1006	JetAirways	Mumbai	Bangalore	2500	<a href="#">Book</a>
1009	JetAirways	Mumbai	Bangalore	3200	<a href="#">Book</a>
1106	SpiceJet	Mumbai	Bangalore	2500	<a href="#">Book</a>
1209	JetAirways	Mumbai	Bangalore	3200	<a href="#">Book</a>
1212	SpiceJet	Mumbai	Bangalore	3000	<a href="#">Book</a>
1902	SpiceJet	Mumbai	Bangalore	3000	<a href="#">Book</a>

#### **4. Form Submission Confirmation:**

- Display a confirmation message to the user before submitting the form to ensure they have reviewed their inputs.

**Congratulations, Your tickets have been booked !**

FLIGHT NUMBER - 1009  
**FLIGHT NAME** - JetAirways  
SOURCE - Mumbai  
DESTINATION - Bangalore  
COST per person - 3200  
  
TOTAL COST - 25600.00

Submit

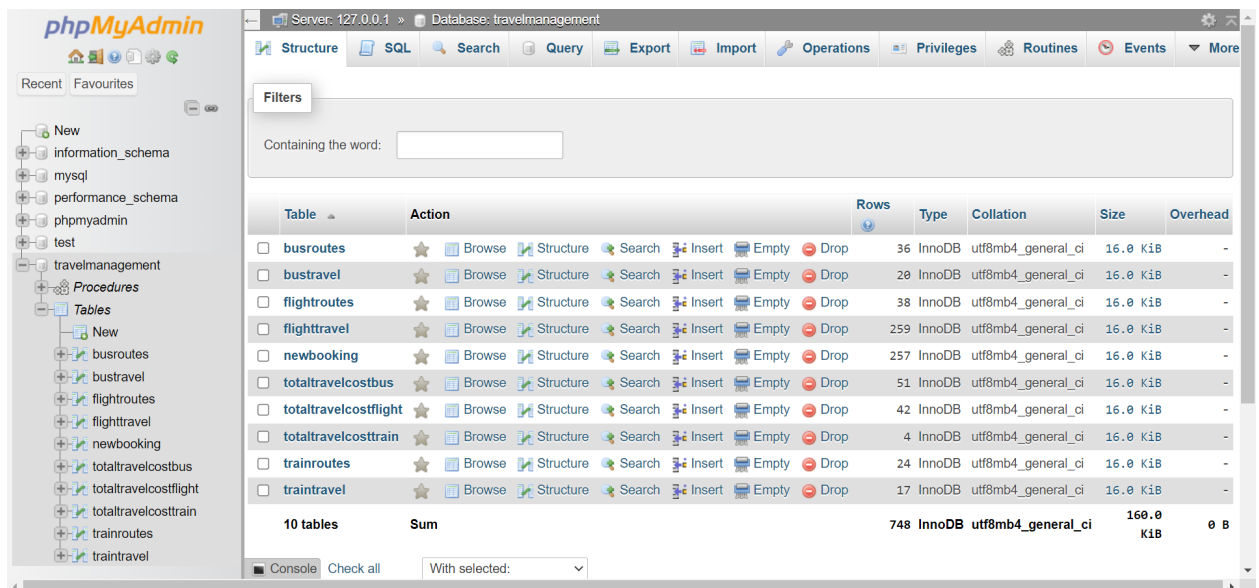
These are just a few examples of client-side validations. The specific validations required for the travel management system project may vary depending on the project requirements and user inputs. It's essential to implement validations that enforce data integrity and improve the overall user experience by providing prompt feedback on any invalid inputs.



## Server-side database handling details

In the travel management system project, server-side database handling is crucial for storing and retrieving data related to user profiles, ticket bookings, and other relevant information.

Here are some key considerations for server-side database handling:



The screenshot shows the phpMyAdmin interface for a database named 'travelmanagement'. The left sidebar displays a tree view of the database structure, including schemas like 'information\_schema', 'mysql', 'performance\_schema', 'phpmyadmin', 'test', and 'travelmanagement'. Under 'travelmanagement', there are 'Procedures' and 'Tables'. The 'Tables' section is expanded, showing a list of tables: 'busroutes', 'bustravel', 'flightroutes', 'flighttravel', 'newbooking', 'totaltravelcostbus', 'totaltravelcostflight', 'totaltravelcosttrain', 'trainroutes', and 'traintravel'. The main panel shows the 'Structure' tab for the 'travelmanagement' database. It lists 10 tables with their respective actions (Browse, Structure, Search, Insert, Empty, Drop) and details (Rows, Type, Collation, Size, Overhead). The tables are all InnoDB with utf8mb4\_general\_ci collation. The total size of the tables is 160.0 KiB.

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> busroutes	★ Browse Structure Search Insert Empty Drop	36	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> bustravel	★ Browse Structure Search Insert Empty Drop	20	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> flightroutes	★ Browse Structure Search Insert Empty Drop	38	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> flighttravel	★ Browse Structure Search Insert Empty Drop	259	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> newbooking	★ Browse Structure Search Insert Empty Drop	257	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> totaltravelcostbus	★ Browse Structure Search Insert Empty Drop	51	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> totaltravelcostflight	★ Browse Structure Search Insert Empty Drop	42	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> totaltravelcosttrain	★ Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> trainroutes	★ Browse Structure Search Insert Empty Drop	24	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> traintravel	★ Browse Structure Search Insert Empty Drop	17	InnoDB	utf8mb4_general_ci	16.0 KiB	-
10 tables	Sum	748	InnoDB	utf8mb4_general_ci	160.0 KiB	0 B



## **2. Database Design:**

- Design the database schema to represent the entities, relationships, and attributes required for the project.
- Identify primary keys, foreign keys, and appropriate data types for each attribute.
- Normalize the database to eliminate redundancy and ensure data integrity.

## **3. Establish Database Connection:**

- Use appropriate programming language (such as PHP) to establish a connection between the server-side code and the database.
- Provide the necessary credentials (database host, username, password, database name) to establish the connection.

## **4. Execute SQL Queries:**

- Use SQL (Structured Query Language) to write and execute queries for database operations such as data insertion, retrieval, updating, and deletion.
- Sanitize user inputs to prevent SQL injection attacks by using prepared statements or parameterized queries.

## **5. Ticket Booking and Management:**

- Implement database operations for ticket bookings, storing relevant details such as ticket type, departure date, arrival date, seat number, fare, and associated user details.
- Retrieve and display booking details for users, manage cancellations or modifications by updating the corresponding database records.

## **6. Database Backup and Maintenance:**

- Regularly back up the database to prevent data loss in case of server failures.
- Implement scheduled tasks for database maintenance, including optimizing queries, indexing, and ensuring data consistency.

## CONCLUSION

In conclusion, the development of a travel management system with integrated train, flight, and bus ticket bookings using HTML, CSS, PHP, and MySQL offers significant benefits for travelers and travel agencies. The project aims to address the existing challenges of fragmented booking processes, limited real-time information, inefficient comparison and selection, payment and ticket management issues, and the lack of personalization.

By implementing a centralized system, users can enjoy a streamlined booking experience where they can easily search and book tickets for trains, flights, and buses from a single platform. Real-time information on availability, schedules, and prices enhances decision-making and minimizes the chances of errors or missed opportunities. The system also allows for efficient payment processing and ticket management, reducing complexities and ensuring smooth transactions.

Additionally, the travel management system enables personalization by offering features like preferred seat selection, special requests, and customized travel itineraries. This enhances user satisfaction and provides a more tailored experience.

By leveraging HTML, CSS, PHP, and MySQL, the project enables seamless integration of the front-end interface, server-side processing, and database management. This ensures the

reliability, scalability, and security of the system.

Overall, the travel management system significantly improves the efficiency and convenience of ticket bookings, enhances user experience, and simplifies the administrative tasks for travel agencies. It promotes a more organized and streamlined approach to travel management, benefiting both travelers and service providers alike.

## APPENDIX

### **a. Tools used:**

For FrontEnd - VSCode (HTML & CSS)

For Backend - MySql

For connecting both - PHP

For Entity Relationship Diagram - Draw.io

### **b. References:**

[HTML Styles CSS \(w3schools.com\)](https://www.w3schools.com/html/html_styles.asp)

[MDN Web Docs \(mozilla.org\)](https://developer.mozilla.org/en-US/docs/Web)

[PHP Tutorial \(w3schools.com\)](https://www.w3schools.com/php/)