## INTRODUCTION

The Tour Management System is a comprehensive project designed to streamline the operations of travel agencies and tour operators. This software facilitates the organization and management of tour packages, bookings, and customer details efficiently. It helps in handling daily transactions related to tour planning, customer registrations, and itinerary management while ensuring easy access to travel details when required.

The system is developed to cater to the diverse needs of the tourism industry, making it easier to manage tour schedules, track reservations, and generate reports on bookings and customer preferences. With this software, travel agencies can efficiently coordinate with hotels, transport providers, and tour guides, ensuring a seamless experience for travelers. Additionally, it helps in maintaining real-time records of available tour packages, customer inquiries, and payments, ensuring smooth operations.

By integrating all essential functionalities, this **Tour Management System** enhances the efficiency of tour operators, simplifies customer handling, and ensures timely execution of travel plans, ultimately improving customer satisfaction.

* 1. **OBJECTIVES**

**The primary objectives of the “Tour Management System” project are:**

* **Efficient Management of Tour Operations**: Streamlining the planning, organization, and execution of tour packages, including itinerary management, customer bookings, and travel arrangements.
* **Ensuring Data Integrity and Security:** Implementing proper access controls and data validation to protect sensitive customer and business information.
* **Simplifying Customer and Booking Management:** Providing a centralized system to manage customer inquiries, bookings, payments, and travel preferences efficiently**.**
* **Enhancing Data Analysis and Reporting**: Enabling administrators to track and analyze tour details, customer preferences, and financial records for better decision-making.
* **Facilitating Easy Communication and Updates:** Allowing seamless updates and management of customer contact information, tour schedules, and travel updates for an improved customer experience.

## 

**2. SYSTEM ANALYSIS**

**2.1 Existing System**

The current system provides basic functionalities for two types of users: Travelers and Tour Operators/Agencies. Travelers can browse available tour packages, make bookings, check the status of their reservations, and update their personal details. Similarly, tour operators and agencies can manage tour listings, track bookings, and update travel itineraries.

The system maintains records of tour packages, customer details, and travel schedules. It may include modules such as **tour package management, customer registration, booking system, and itinerary planning**. However, the existing system may lack automation, real-time updates, and an efficient way to handle customer inquiries and payments, making it necessary to implement a more advanced **Tour Management System**.

**2.1.1 DISADVANTAGES**

* **Security Vulnerabilities:** If the system is not properly secured, it could be vulnerable to attacks such as SQL injection and cross-site scripting, potentially compromising customer data, payment details, and travel itineraries.
* **Maintenance Challenges**: Over time, the system may require **updates, bug fixes, or** **new feature implementations**. If the development team lacks expertise in the required technologies, maintaining and upgrading the system can become difficult.
* **Integration Issues**: If the tour management system needs to integrate with third-party services such as payment gateways, hotel booking systems, or airline databases, there may be compatibility and technical challenges that need to be addressed for seamless functionality..

**2.2 PROPOSED SYSTEM**

The **Personalized Tour Planner Software** is designed to cater to individual travelers, travel agencies, and tour operators, offering customized travel experiences. It covers all aspects of personalized tour planning, including user preferences, destination selection, itinerary creation, accommodation and transportation bookings, activity scheduling, and budget management. The system allows users to input their travel interests, preferred dates, and budget to generate tailored tour packages. Additionally, it manages booking confirmations, payment processing, real-time updates, and customer support, ensuring a seamless and hassle-free travel experience.

**2.1.2 ADVANTAGES**

* **Efficient Data Management**: A PHP and MySQL-based system can handle large amounts of travel-related data, including user preferences, bookings, destinations, and itineraries, ensuring seamless organization and quick access.
* **Enhanced Travel Planning**: The system provides real-time updates on availability, pricing, and travel options, helping users make informed decisions and customize their trips efficiently.
* **Automated Itinerary Generation**: Based on user inputs, the system can automatically generate tailored travel itineraries, saving time and reducing the hassle of manual trip planning.
* **Seamless Booking and Payment Processing**: Integrated booking and payment features allow users to reserve accommodations, transportation, and activities in one place, ensuring a smooth and secure transaction process.
* **Data Analysis and Reporting:** The system can generate detailed reports on user preferences, booking trends, and travel patterns, helping agencies and operators refine their services and improve customer satisfaction.

**2.3 SYSTEM STUDY**

**2.3.1 FEASIBILITY ANALYSIS**

Feasibility analysis evaluates the impact of developing a **Personalized Trip** **System** on users and businesses. The impact can be either positive or negative. If the benefits outweigh the challenges, the system is considered feasible. The feasibility study is conducted in two key areas:

The three key considerations involved in the feasibility analysis are:

* **Economic Feasibility**
* **Technical Feasibility**

**2.3.2 TECHNICAL FEASIBILITY**

The development and maintenance of the **Personalized Trip Planner System** do not require significant additional resources. All necessary tools, technologies, and expertise for software development are readily available within the organization. The system can be built using existing infrastructure, ensuring smooth development and integration. **ECONOMICAL FEASIBILITY**

The development of this application is highly economically feasible. The organization does not need to invest heavily in new resources, as most of the required technology and infrastructure are already available. The main requirement is to create a structured environment for development with proper supervision. Once implemented, the system will continue to function efficiently without requiring substantial ongoing investments, making it a cost-effective solution.

## 

## 3. SYSTEM SPECIFICATION

**3.1 HARDWARE REQUIREMENTS**

Processor **:** AMD PRO A4-3350B R4

Processor Speed **:** 2.00GHz Onwards

RAM **:** 4 GB

Hard Disk **:** 500 GB

Monitor **:** LG 23”

Network card **:** Any card can provide a 100mbps speed

**3.2 SOFTWARE REQUIREMENTS**

Operating system **:** Windows 7**/**8**/**10

Front End **:** PHP Framework

Back End **:** MySQL

UI Design **:** CSS, Bootstrap

Development Tool **:** XAMPP Server

**4. SOFTWARE DESCRIPTION**

**4.1 INTRODUCTION TO FRONTEND**

**HTML Overview**

Hyper Text Markup Language (HTML) is the standard markup language for creating web pages and web applications**.** With Cascading Style Sheets (CSS), and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a webserver or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document**.**

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects, such as interactive forms may be embedded into the rendered page**.** It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img> and <input /> introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page**.**

HTML can embed programs written in a scripting language such as JavaScript which affect the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997**.**

**Development**

In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners- Lee wrote a memo proposing an Internet-based hypertext system Berners-Lee specified HTML and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes from 1990 he listed "some of the many areas in which hypertext is used" and put an encyclopedia first**.**

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Tim Berners-Lee in late 1991.It describes 18 elements comprising the initial, relatively simple design of HTML. Except for the hyperlink tag, these were strongly influenced by SGMLguid, an in-house Standard Generalized Markup Language (SGML)-based documentation format at CERN. Eleven of these elements still exist in HTML 4**.**

HTML is a markup language that web browsers use to interpret and compose text, images, and other material into visual or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS. Many of the text elements are found in the 1988 ISO technical report TR 9537 Techniques for using SGML, which in turn covers the features of early text formatting languages such as that used by the RUNOFF command developed in the early 1960s for the CTSS (Compatible Time-Sharing System) operating system: these formatting commands were derived from the commands used by typesetters to manually format documents. However, the SGML concept of generalized markup is based on elements rather than merely print effects, with also the separation of structure and markup; HTML has been progressively moved in this direction with CSS**.**

Berners-Lee considered HTML to be an application of SGML. It was formally defined as such by the Internet Engineering Task Force (IETF) with the mid-1993 publication of the first proposal for an HTML specification: "Hypertext Markup Language (HTML)" Internet-Draft by Berners-Lee and Dan Connolly, which included an SGML Document Type Definition to define the grammar. The draft expired after six months, but was notable for its acknowledgment of the NCSA Mosaic browser's custom tag for embedding in-line images, reflecting the IETF's philosophy of basing standards on successful prototypes. Similarly, Dave Raggett's competing Internet-Draft, "HTML+ (Hypertext Markup Format)", from late 1993, suggested standardizing already-implemented features like tables and fill-out forms**.**

**Markup**

HTML markup consists of several key components, including those called tags (and their attributes), character-based data types, character references and entity references**.** HTML tags most commonly come in pairs like <h1> and </h1>, although some represent empty elements and so are unpaired, for example <img>**.** The first tag in such a pair is the start tag, and the second is the end tag (they are also called opening tags and closing tags)**.** Another important component is the HTML document type declaration, which triggers standards mode rendering. The following is an example of the classic Hello world program, a common test employed for comparing programming languages, scripting languages and markup languages**.**

This example is made using 9 lines of code:

<!DOCTYPE html>

**<html>**

**<head>**

**<title>**This is a title**</title>**

**</head>**

**<body>**

**<p>**Hello world!**</p>**

**</body>**

**</html>**

**Elements**

HTML documents imply a structure of nested HTML elements. These are indicated in the document by HTML tags, enclosed in angle brackets thus: <p>**.** In the simple, general case, the extent of an element is indicated by a pair of tags: a "start tag" <p> and "end tag" </p>. The text content of the element, if any, is placed between these tags. Tags may also enclose further tag markup between the start and end, including a mixture of tags and text. This indicates further (nested) elements, as children of the parent element. The start tag may also include attributes within the tag. These indicate other information, such as identifiers for sections within the document, identifiers used to bind style information to the presentation of the document, and for some tags such as the <img> used to embed images, the reference to the image resource.

**CSS Overview**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML 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 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. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified. Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

**sources**

CSS information can be provided from various sources. These sources can be the web browser, the user and the author. The information from the author can be further classified into inline, media type, importance, selector specificity, rule order, inheritance and property definition. CSS style information can be in a separate document or it can be embedded into an HTML document. Multiple style sheets can be imported. Different styles can be applied depending on the output device being used; for example, the screen version can be quite different from the printed version, so that authors can tailor the presentation appropriately for each medium.

The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. This process is called cascading. One of the goals of CSS is to allow users greater control over presentation. Someone who finds red italic headings difficult to read may apply a different style sheet. Depending on the browser and the web site, a user may choose from various style sheets provided by the designers, or may remove all added styles and view the site using the browser's default styling, or may override just the red italic heading style without altering other attributes.

**Example**

Consider this HTML fragment**:**

<!DOCTYPE html>

**<html>**

**<head>**

<**meta** charset="utf-8">

**<style>**

**#xyz** {color: red; }

**</style>**

**</head>**

**<body>**

<**p** id="xyz" style="color: blue;"> To demonstrate specificity

**</p>**

**</body>**

**</html>**

In the above example, the declaration in the style attribute overrides the one in the <style> element because it has a higher specificity.

**Browser support**

Each web browser uses a layout engine to render web pages, and support for CSS functionality is not consistent between them. Because browsers do not parse CSS perfectly, multiple coding techniques have been developed to target specific browsers with workarounds (commonly known as CSS hacks or CSS filters). Adoption of new functionality in CSS can be hindered by lack of support in major browsers. For example, Internet Explorer was slow to add support for many CSS 3 features, which slowed adoption of those features and damaged the browser's reputation among developers.In order to ensure a consistent experience for their users, web developers often test their sites across multiple operating systems, browsers, and browser versions, increasing development time and complexity. Tools such as Browser Stack have been built to reduce the complexity of maintaining these environments.

In addition to these testing tools, many sites maintain lists of browser support for specific CSS properties, including Can I Use and the Mozilla Developer Network. Additionally, the CSS 3 defines feature queries, which provide an @supports directive that will allow developers to target browsers with support for certain functionality directly within their CSS. CSS that is not supported by older browsers can also sometimes be patched in using Javascript polyfills, which are pieces of Javascript code designed to make browsers behave consistently. These workarounds-and the need to support fallback functionality- can add complexity to development projects, and consequently, companies frequently define a list of browser versions that they will and will not support.

**Vertical Control limitations**

Though horizontal placement of elements was always generally easy to control, vertical placement was frequently unintuitive, convoluted, or outright impossible. Simple tasks, such as centering an element vertically or placing a footer no higher than bottom of the viewport required either complicated and unintuitive style rules, or simple but widely unsupported rules. The Flexible Box Module improved the situation considerably and vertical control is much more straightforward and supported in all of the modern browsers. Older browsers still have those issues, but most of those (mainly Internet Explorer 9 and below) are no longer supported by their vendors.

**Absence of expressions**

There was no standard ability to specify property values as simple expressions (such as margin-left: 10% 3em + 4px;). This would be useful in a variety of cases, such as calculating the size of columns subject to a constraint on the sum of all columns. Internet Explorer versions 5 to 7 support a proprietary expression() statement, with similar functionality. This proprietary expression() statement is no longer supported from Internet Explorer 8 onwards, except in compatibility modes. This decision was taken for "standards compliance, browser performance, and security reasons".However, a candidate recommendation with a calc() value to address this limitation has been published by the CSS WG and has since been supported in all of the modern browsers.

**PHP**

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team.PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications. The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification.

**History**

PHP development began in 1995 when Rasmus Lerdorf wrote several Common Gateway Interface (CGI) programs in C,which he used to maintain his personal homepage. He extended them to work with web forms and to communicate with databases, and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI.

PHP/FI could help to build simple, dynamic web applications. To accelerate bug reporting and to improve the code, Lerdorf initially announced the release of PHP/FI as "Personal Home Page Tools (PHP Tools) version 1.0" on the Usenet discussion group comp.infosystems. www.authoring.cgi on June 8, 1995. This release already had the basic functionality that PHP has as of 2013. This included Perl-like variables, form handling, and the ability to embed HTML. The syntax resembled that of Perl but was simpler, more limited and less consistent.

Lerdorf did not intend the early PHP to become a new programming language, but it grew organically, with Lerdorf noting in retrospect: "I don't know how to stop it, there was never any intent to write a programming language. I have absolutely no idea how to write a programming language, I just kept adding the next logical step on the way. "A development team began to form and, after months of work and beta testing, officially released PHP/FI 2 in November 1997.

The fact that PHP lacked an original overall design but instead developed organically has led to inconsistent naming of functions and inconsistent ordering of their parameters. In some cases, the function names were chosen to match the lower-level libraries which PHP was "wrapping", while in some very early versions of PHP the length of the function names was used internally as a hash function, so names were chosen to improve the distribution of hash values.

**Syntax**

The following "Hello, World!" program is written in PHP code embedded in an HTML document:

<!DOCTYPE html>

**<html>**

**<head>**

**<title>**PHP Test**</title>**

**</head>**

**<body>**

<?**php echo** '<p>Hello World</p>'; ?>

**</body>**

**</html>**

The PHP interpreter only executes PHP code within its delimiters. Anything outside its delimiters is not processed by PHP, although non-PHP text is still subject to control structures described in PHP code. The most common delimiters are <?php to open and ?> to close PHP sections. The shortened form <? also exists. This short delimiter makes script files less portable, since support for them can be disabled in the local PHP configuration and it is therefore discouraged. However, there is no recommendation against the use of the echo short tag <?=Prior to PHP 5.4.0, this short syntax for echo() only works with the short\_open\_tag configuration setting enabled, while for PHP 5.4.0 and later it is always available. The purpose of all these delimiters is to separate PHP code from non-PHP content, such as JavaScript code or HTML markup. The first form of delimiters, <?php and ?>, in XHTML and other XML documents, creates correctly formed XML processing instructions. This means that the resulting mixture of PHP code and other markup in the server-side file is itself well-formed XML.

Variables are prefixed with a dollar symbol, and a type does not need to be specified in advance. PHP 5 introduced type hinting that allows functions to force their parameters to be objects of a specific class, arrays, interfaces or call back functions. However, before PHP 7.0, type hints could not be used with scalar types such as integer or string.

Unlike function and class names, variable names are case sensitive. Both double- quoted ("") and here doc strings provide the ability to interpolate a variable's value into the string. PHP treats newlines as whitespace in the manner of a free-form language, and statements are terminated by a semicolon.PHP has three types of comment syntax: /\*\*/ marks block and inline comments; // as well as # are used for one-line comments. The echo statement is one of several facilities PHP provides to output text, e.g., to a web browser. In terms of keywords and language syntax, PHP is similar to the C style syntax. if conditions, for and while loops, and function returns are similar in syntax to languages such as C, C++, C#, Java and Perl.

**4.2 INTRODUCTION TO BACKEND**

**MySQL**

MySQL is an open-source relational database management system (RDBMS).Its name is a combination of "My", the name of co-founder Michael Widenius' daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is a central component of the LAMP open-source web application software stack (and other "AMP" stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python". Applications that use the MySQL database include: TYPO3, MODx, Joomla, WordPress, phpBB, MyBB, and Drupal. MySQL is also used in many high-profile, large-scale websites, including Google (though not for searches), Facebook, Twitter, Flickr, and YouTube.

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, OS X, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64, A port of MySQL to OpenVMS also exists.

**History**

MySQL was created by a Swedish company, MySQL AB, founded by David Axmark, Allan Larsson and Michael "Monty" Widenius. Original development of MySQL by Widenius and Axmark began in 1994. The first version of MySQL appeared on 23 May 1995. It was initially created for personal usage from mSQL based on the low-level language ISAM, which the creators considered too slow and inflexible. They created a new SQL interface, while keeping the same API as mSQL. By keeping the API consistent with the mSQL system, many developers were able to use MySQL instead of the (proprietarily licensed) mSQL antecedent.

The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, beginning from 28 June 2000 (which in 2009 has been extended with a FLOSS License Exception) or to use a proprietary license.Support can be obtained from the official manual. Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services, including MariaDB and Percona.

MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case" and that the "developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is very, very good". It has also been tested to be a "fast, stable and true multi-user, multi-threaded sql database server".

**Features**

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

* A broad subset of ANSI SQL 99, as well as extension
* Cross-platform support
* Stored procedures, using a procedural language that closely adheres to SQL/PSM
* Triggers
* Cursors
* Updatable views
* Online DDL when using the InnoDB Storage Engine.
* Information schema
* Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
* A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
* X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine
* Transactions with save points when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
* ACID compliance when using InnoDB and NDB Cluster Storage Engines
* SSL support
* Query caching
* Sub-SELECTs (i.e. nested SELECTs)
* Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.Multi-master replication is provided in MySQL Cluster, and multi-master support can be added to unclustered configurations using Galera Cluster
* Full-text indexing and searching
* Embedded database library
* Unicode support
* Partitioned tables with pruning of partitions in optimizer
* Shared-nothing clustering through MySQL Cluster
* Multiple storage engines, allowing one to choose the one that is most effective for each table in the application
* Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.

**5. SYSTEM DESIGN**

**5.1 MODULE DESCRIPTION**

* **User Management & Authentication**
* **Inventory management**
* **Booking & Tour Management**
* **Payments**
* **Admin & System Management**

**User Management & Authentication**

This module handles user registration, authentication, and role-based access control. It ensures that users, whether customers or administrators, can securely log in, manage their profiles, and perform actions based on their roles.

**Inventory management**

This module manages the availability of tours, travel packages, accommodations, and transport options. It ensures that only available tours are bookable and tracks resource usage.

**Booking & Tour Management**

This module manages the availability of tours, travel packages, accommodations, and transport options. It ensures that only available tours are bookable and tracks resource usage.

**Payments**

This module handles financial transactions, ensuring secure and accurate payment processing for tour bookings. It may integrate with payment gateways.

**Admin & System Management**

This module provides administrators with tools to oversee system operations, manage users, and configure tour details. It ensures smooth backend functionality.

**5.2 ARCHITECTURE DESIGN**

PERSONALIZED TRIP PLANNER

Administrator

Users

System management

Login / Logout

View Doners

Manage Blood and Doners

Contect Both Admin and Doners

Add Blood and Category management

Request

All data’s management

**5.3 DATA FLOW DIAGRAM**

Admin

ANANDA BLOOD LINE

Recipient

Donor

**Fig 5.3 DFD Level 0**

Recieve Blood Health Information

Blood info

Recieve Request

Blood Request

Confirm Donation **/** Request Blood claim Confirmation

Recipient

Manage Recipient Information

Manage Blood Bank Transactions

Manage Donation Information

Donor

Admin

**Fig 5.3 DFD Level 1**

Donation Confirmation

Donor info

Donation Confirmation Donor info

New Donor View Recipient

Applicable as Recipient View Donor

Request Confirmation Blood Request

Blood Request Request Confirmation

## 

## ER DAIGRAM

|  |  |
| --- | --- |
| Blood Donors | |
| String | Id |
| String | Name |
| String | Mobile no |
| String | Email |
| String | Pass |
| String | Gender |
| String | Age |
| String | Blood Group |
| String | Address |
| String | Department |

|  |  |
| --- | --- |
| Blood Group | |
| String | id |
| String | Blood Group |

## 

Admin

|  |  |
| --- | --- |
| Contect us | |
| String | Id |
| String | Address |
| String | Email |
| String | Contect no |

|  |  |
| --- | --- |
| Blood Requir | |
| String | id |
| String | Blood Donor id |
| String | Name |
| String | Email |
| String | Contect no |
| String | Blood requir |
| String | date |

## 

**5.4 TABLE DESIGN**

**Table of Admin Login**

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| Name | Varchar(120) |  |
| psw | varchar(120) | NULLABLE |

## Table of Book Table

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| id | int(50) |  |
| uid | varchar(100) | NULLABLE |
| rid | varchar(50) | NULLABLE |
| sdate | varchar(100) | NULLABLE |
| edate | varchar(50) | NULLABLE |
| amnt | varchar(50) | NULLABLE |
| tdays | varchar(50) | NULLABLE |
| tamnt | varchar(50) | NULLABLE |
| status | varchar(50) | NULLABLE |

## Table of Ticket Booking

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| id | varchar(50) |  |
| uid | varchar(50) | NULLABLE |
| pid | varchar(50) | NULLABLE |
| np | varchar(50) | NULLABLE |
| status | varchar(50) | NULLABLE |
| amnt | varchar(50) | NULLABLE |

## Tablo of Feedback

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| ID | int(50) |  |
| uid | varchar(50) | NULLABLE |
| feedback | varchar(255) | NULLABLE |

## Table of Package

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| id | int(50) |  |
| pname | varchar(50) | NULLABLE |
| package\_price | varchar(255) | NULLABLE |
| place | varchar(255) | NULLABLE |
| sdate | varchar(50) | NULLABLE |
| edate | varchar(50) | NULLABLE |
| img | varchar(50) | NULLABLE |
| des | varchar(250) | NULLABLE |

## Table of Register Info

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| id | int(50) |  |
| name | varchar(50) | NULLABLE |
| gender | varchar(50) | NULLABLE |
| age | varchar(50) | NULLABLE |
| varchar(50) | varchar(50) | NULLABLE |
| phone | varchar(50) | NULLABLE |
| location | varchar(50) | NULLABLE |
| address | varchar(200) | NULLABLE |
| uname | varchar(50) | NULLABLE |
| psw | varchar(50) | NULLABLE |

## Table of Transport

|  |  |  |
| --- | --- | --- |
| Field | Type | Other |
| id | int(5) |  |
| vname | varchar(50) | NULLABLE |
| seats | varchar(50) | NULLABLE |
| vno | varchar(50) | NULLABLE |
| amnt | varchar(50) | NULLABLE |

1. **SYSTEM IMPLEMENTAION**

**SYSTEM IMPLEMENTAION**

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over, an evaluation of change over methods. Apart from planning major task of preparing the implementation are education and training of users. The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out, discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system.

In network backup system no additional resources are needed. Implementation is the final and the most important phase. The most critical stage in achieving a successful new system is giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and if it is found to be working according to the specification. This method also offers the greatest security since the old system can take over if the errors are found or inability to handle certain type of transactions while using the new system.

As the part of system testing we execute the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. The ultimate aim is quality assurance. Tests are carried out and the results are compared with the expected document. In the case of erroneous results, debugging is done. Using detailed testing strategies a test plan is carried out on each module. The various tests performed are unit testing, integration testing and user acceptance testing.

**7. SYSTEM TESTING**

**7.1. UNIT TESTING**

The software units in the system are modules and routines that are assembled and integrated to perform a specific function. As a part of unit testing we executed the program for individual modules independently. This enables, to detect errors in coding and logic that are contained within each of the three modules. This testing includes entering data that is filling forms and ascertaining if the value matches to the type and entered into the database. The various controls are tested to ensure that each performs its action as required.

Unit testing verification efforts on the smallest unit of software design, module. This is known as "module testing". After testing each every field in the modules, the modules of the project is tested separately. Unit testing focuses verification efforts on the smallest unit of software design and field. For example, username and password are entered in correct manner and checked. While filling the details in the register form certain fields are left as empty and checked. The submit button successfully stores the data in the databases. This is done for each and every module individually.

"Unit testing" is a software testing method testing method by which individual unit of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedure, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short fragments created by programmers or occasionally by white box testers during the development process. It is also known as component testing.

Ideally, each test case is independent from the other. Substitutes such as method stubs, mock object, fakes and test harness can be used to assist testing a module in isolation, unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended. The primary goal of unit testing is to take the smallest piece of testable software in the application, isolation it from the reminder of the code, and determine whether it behaves exactly as you expect. Each unit tested separately before integrating them

into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use.

The most common approach to unit testing requires drivers and stubs to be written, the driver simulates a calling unit and the stub simulates a called unit. The investment of developer in this activity sometimes results in demoting unit testing to a lower level of priority and that is almost always a mistake. Even though the drivers and stubs cost time and money, unit testing provides some undeniable advantages. It allows for automation of the testing process, reduce difficulties of discovering errors contained in more complex pieces of the application, and test coverage is often enhanced because attention is given to each unit. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the pieces of code must satisfy. As a result, it affords several benefits.

**Finds Problem Early:**

Unit testing finds problems early in the development cycle. In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the test. Since the unit tests alert the development team of the problem before handling the code off to testers or clients, it is still early in the development process.

**Facilitates Change:**

Unit testing allows the programmer to re factor code at a later date, and make sure the module still works correctly. The procedure is to write test cases for all functions and methods so that whenever a change a fault, it can be quickly identified. Readily available unit tests make it easy for the programmer to check whether a piece of code is still working properly. In continuous unit testing environments, through the inherent practice of sustained maintenance, unit tests will continue to accurately reflect the intended use of the executable and code in the face of any change. Depending upon established development practices and unit test coverage, up-to-the-second accuracy can be maintained.

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data. Usage procedures, and operating procedures, are tested to determine whether they fit for use. Intuitively one can view a unit as the smallest testable part of an application. Unit testing is way of testing software components. The "unit" is the thing being tested. You can do both black and white box testing with unit tests; the concept is orthogonal to white/black-box testing.

**White Box Testing:**

Structured testing is known as "white box testing" or "glass box testing" program errors can be classified as missing path errors, computational errors and domain errors. "white-box testing" can be applied at the unit, integration and system level of the software testing process. Although traditional tester's tenders to think of white box testing as being done at the unit level, it is used integration more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Though this of design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specifications or missing requirements.

**Black Box Testing:**

Stress tests drive the system to its limits. They are design to internationally break the unit. Structure tests verify logical execution paths. Functional, performance and stress tests are collectively known as "black box testing"

"black-box" testing is a method of software testing that examines the functionally of an application without peering into its internal structure or workings.

This method of test can be applied to virtually every level of software testing:

* Unit, integration, system and acceptance it typically comprises most if not all higher level testing, but can also dominate unit testing as well
* In black box testing, you don't care how the internals of the thing being tested work. You invoke the exposed API and check the result; you don't care what to the thing being tested did to give you the result.
* In white box testing, you don't care how the internal of the thing being tested work. So instead of just checking the output of your thing, you might check that internal variables to the thing being tested are correct.
* Unit testing is a way of testing software components. The "unit" is the thing being tested. You can do both black and white box testing with unit tests; the concept is orthogonal to white/black-box testing.

**7.2. INTEGRATION TESTING**

Integration testing is done to take unit-tested modules and build a program structure that has dictated by design. All the modules were integrated after the completion of unit test. The modules are integrated by moving downward through, beginning with the main module.

After the successful integration of the modules, the system was found to be running with no error, here, on clicking the submit button the detail are updated on the data base, also an email with the user name and password is sent to the user's mail ID. Similarly transaction ID is generated, stored in the data base and it sent to the user's mail ID. Thus all the modules are integrated and are tested successfully. Integration testing is a phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and after validation testing. Integration takes as its input modules that have been unit tested, groups them in larger aggregates.

**Big Bang:**

In this approach, all or most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing. The big bang method is very effective for saving time in the integration testing process. However if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing.

A type of big bang integration testing is called usage model testing. Usage model testing can be used in both software and hardware integration testing. The basic behind this type of integration testing is to run user-like workloads in integrated user-like environment. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the environment. For integration testing.

Usage model testing can be more efficient and provides better test coverage than traditional focused functional testing. To more efficient and accurate, care must be used in defining the user-like workloads for creating realistic environment will work as expected for the target customers.

**Simplifies Integration:**

Integration testing may reduce uncertainly in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier. An elaborate In integration hierarchy of unit tests does not equal integration testing. Integration with peripheral units should be included in integration tests, but not in unit test integration testing typically still relies heavily on humans testing manually; high-level or global-scope testing can be difficult to automate, such that manual testing often appears faster and cheaper.

**Top-Down and Bottom-Up:**

**Bottom-Up Testing:**

"Bottom up testing" is approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested.

All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower integrated modules, the next level of modules will be formed and can be used for integration testing This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.

**Top-Down Testing:**

Top down testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module.

**Sandwich Testing:**

Sandwich testing is an approach to combine top down testing with bottom up testing The main advantage of the bottom-up approach is that bugs are more easily found. With top- down, it is easier to find a missing branch link.

**7.3. USER ACCEPTANCE TESTING**

Validation testing provides the assurance that software needs all the functional behavioural and performance requirements. Validation testing can be defined as the software functions in a manner that is expected by the user. This testing verifies that all elements combine properly and that overall system function and performance is achieved. After the integration of the modules, the validation test was carried out over by the system. It was found that all the modules work well together and meet the overall system function and performance.

Validation testing provides a sort of living documentation of the system. Develops looking to learn what functionally provided by a validation, and how to use it, can look at the unit tests to gain a basic understanding of the validation interface. Validation are independent that are used together for checking that a product, services, or system meets requirement and specification and that it fulfils its ISO 9000. The words "verification" and "validation" are sometimes preceded with "independent", indicating that the verification and validation is to be performed by a disinterested third party. "Independent verification and validation" can be abbreviated as "IV&V".

"Validation The assurance that a product, service, or system meets the needs of the customer and other identified stakeholder. It often involves acceptance and suitability with external customer. Contrast with verification". The evaluation of whether or not a product, service or system complies with a regulation, requirement, specification, or imposed condition. It is often as internal process. Contrast with validation

"Verification is intended to check that a product service, or system (or portion thereof, or set thereof) meets a set of design specification. In the development phase, verification procedures involve performing special tests to model or simulates a portion, or the entirely, service or system, then performing a review or analysis of the modelling result. In the post-development phase, verification procedures involve regularly repeating tests devised specifically to ensure that the product, service, or system continues to meet the initial design requirement, specification, and regulation as time progresses it is a process that is used to evaluate whether a product, service or system complies with regulations, specification, or condition imposed at the start of a development phase. Verification can be in development, scale-up, or production. This is often a internal process.

**8. CONCLUSION**

The **Personalized Trip Planner** is a dynamic and user-centric web application designed to streamline the travel planning experience. Developed using **PHP and MySQL**, the system enables users to effortlessly **book trips, schedule itineraries, and process payments** in a secure and efficient manner. The integration of **real-time booking features** ensures a smooth experience, while personalized recommendations enhance user satisfaction. Additionally, the platform offers **admin functionalities** to manage users, bookings, and payments, making it a comprehensive solution for both travel and service providers. Future enhancements, such as **AI-powered travel suggestions, real-time availability updates, multi-language support, and mobile app integration**, can significantly improve accessibility and engagement. By continuously refining its features and incorporating modern technologies, the **Personalized Trip Planner** serves as a robust foundation for creating smarter, more intuitive, and customized travel experiences for users worldwide.

**FUTURE ENHANCEMENTS:**

* **User Experience**
* **Interactive Maps:** Integrate Google Maps or OpenStreetMap for real-time route planning.
* **AI-Powered Recommendations:** Suggest attractions, restaurants, and activities based on user preferences.
* **Multi-Language Support:** Allow users to select different languages.
* **Booking & Payments**
* **Real-Time Availability:** Sync with hotel and tour operators' APIs for live booking updates.
* **Multiple Payment Options:** Support PayPal, Stripe, and cryptocurrency payments.
* **Dynamic Pricing:** Show price variations based on demand and season.

1. **BIBLIOGRAPHY**

**Book Reference:**

1. **Luke Willing &Laura Thomson ,***"PHP and MySQL Web Development"*, 5th Edition, Addison-Wesley, 2016.
2. **Robin Nixson ,***"Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5"*, 5th Edition, O'Reilly Media, 2018.
3. **Ramez Elmasri &Shamkant B .Navathe ,***"Fundamentals of Database Systems"*, 7th Edition, Pearson, 2016.

**Website Reference:**

* www.w3schools.com/php/default.asp
* www.php.net/manual/en/tutorial.php
* www.phptpoint.com

1. **APPENDIX**
2. **SAMPLE SOURCE CODE**

**HOME PAGE**

<html|>

<title>Tours & Travels</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:180px;

background:url("images/4.png");

background-repeat: no-repeat;

background-size: 100% 300px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<div id="navbar"><p>Tours And Travels</p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a class="active" href="index.php">Home Page</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="tour/admin.php">Admin login</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="tour/user.php">User Login</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="register.php">User Registration</a></li>

</ul>

<div id="bg1"> </div>

<div> &nbsp;</div>

<br>

<br>

<br>

<br>

<br>

<br>

<br>

<br>

<br>

<br>

<br>

<br>

<div id="footer"> copyrights & designedby@Tours & travels</div>

**SIGN UP PAGE**

<?php

include("dbconnect.php");

extract($\_POST);

session\_start();

if(isset($\_POST['btn']))

{

//checking name

$qry=mysqli\_query($conn,"insert into register(name,gender,age,email,phone,location,address,uname,psw) values('$name','$gender','$age','$email','$phone','$loc','$address','$uname','$psw')");

if($qry)

{

echo "<script>alert('inserted sucessfully')</script>";

}

else

{

echo "failed";

}

}

?>

<html>

<title>Temple Seva</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

text-color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:130px;

background:url("images/4.png");

background-repeat: no-repeat;

background-size: 1420px 300px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<script>

function name()

{

var name=/^[a-zA-Z ]{3,20}$/;

if(document.f1.name.value.search(name)==-1)

{

alert("enter correct name");

document.f1.name.focus();

return false;

}

}

function age()

{

var age=/^[0-9]{1,3}$/;

if(document.f1.age.value.search(age)==-1)

{

alert("enter correct age");

document.f1.age.focus();

return false;

}

}

function phone()

{

var phone=/^[0-9]{10}$/;

if(document.f1.phone.value.search(phone)==-1)

{

alert("enter correct phone no");

document.f1.phone.focus();

return false;

}

}

function email()

{

var email=/^[a-zA-Z0-9-\_\.]+@[a-zA-Z]+\.[a-zA-Z]{2,3}$/;

if(document.f1.email.value.search(email)==-1)

{

alert("enter correct email");

document.f1.email.focus();

return false;

}

}

function address()

{

var address=/^[a-zA\_Z0-9 ,#]{5,100}$/;

if(document.f1.address.value.search(address)==-1)

{

alert("enter correct address");

document.f1.address.focus();

return false;

}

}

function loc()

{

var loc=/^[a-zA-Z ]{3,20}$/;

if(document.f1.loc.value.search(loc)==-1)

{

alert("enter correct Location");

document.f1.loc.focus();

return false;

}

}

function uname()

{

var uname=/^[a-zA\_Z0-9]{3,100}$/;

if(document.f1.uname.value.search(uname)==-1)

{

alert("enter correct name");

document.f1.uname.focus();

return false;

}

}

function psw()

{

function vali()

{

var name=/^[a-zA-Z ]{3,20}$/;

var age=/^[0-9]{1,3}$/;

var phone=/^[0-9]{10}$/;

var email=/^[a-zA-Z0-9-\_\.]+@[a-zA-Z]+\.[a-zA-Z]{2,3}$/;

var loc=/^[a-zA-Z ]{3,20}$/;

var address=/^[a-zA\_Z0-9 ,#]{5,100}$/;

var psw=/^[a-zA\_Z0-9]{3,100}$/;

var uname=/^[a-zA\_Z0-9]{3,100}$/;

//var mesg=/^[a-zA\_Z0-9]{10,300}$/;

if(document.f1.name.value.search(name)==-1)

{

alert("enter correct name");

document.f1.name.focus();

return false;

}

else if(document.f1.age.value.search(age)==-1)

{

alert("enter correct Age");

document.f1.age.focus();

return false;

}

else if(document.f1.phone.value.search(phone)==-1)

{

alert("enter correct phone no");

document.f1.phone.focus();

return false;

}

else if(document.f1.email.value.search(email)==-1)

{

alert("enter correct email");

document.f1.email.focus();

return false;

}

else if(document.f1.loc.value.search(loc)==-1)

{

alert("enter correct Location");

document.f1.loc.focus();

return false;

}

else if(document.f1.address.value.search(address)==-1)

{

alert("enter correct address");

document.f1.address.focus();

return false;

}

if(document.f1.uname.value.search(uname)==-1)

{

alert("enter correct user name");

document.f1.uname.focus();

return false;

}

else if(document.f1.psw.value.search(psw)==-1)

{

alert("enter correct password");

document.f1.psw.focus();

return false;

}

else

{

return true;

}

}

</script>

<div id="navbar"><p>Tours And Travels </p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="index.php">Home Page</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="admin.php">Admin login</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="user.php">User Login</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a class="active" href="../register.php">User Registration</a></li>

</ul>

<div id="bg1"> </div>

<form id="f1" name="f1" method="post" action="#" onSubmit="return vali()">

<table width="100%" border="0" align="center">

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td width="10%">&nbsp;</td>

<td width="20%">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td width="11%">&nbsp;</td>

<td width="25%">&nbsp;</td>

<br />

<td colspan="2" align="center" ><div class="style5"><h3>New User Registation</h></div></td>

<td width="23%">&nbsp;</td>

<td width="11%">&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="31">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Name</span></td>

<td><label>

<input name="name" type="text" id="name" onChange="return name ()" required/>

</label></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="30">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Gender</span></td>

<td><input name="gender" type="radio" value="male" required/>

Male

<input name="gender" type="radio" value="female" />

Female</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="34">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Age</span></td>

<td><label>

<input name="age" type="text" id="age" onChange="return age ()" required/>

</label></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="32">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Email Id </span></td>

<td><input name="email" type="text" id="email" onChange="return email()" required/></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Phone Number </span></td>

<td><input name="phone" type="text" id="phone" onChange="return phone()" required/></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Location</span></td>

<td><input type="text" name="loc" id="loc" required></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Address</span></td>

<td><textarea name="address" id="address" required></textarea></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="31">&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">User Name</span></td>

<td><input name="uname" type="text" id="uname" required/></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td><span class="style6">Password</span></td>

<td><input name="psw" type="password" id="psw" required/></td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td height="35">&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<tr>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td><input name="btn" type="submit" id="btn" value="Submit" />

<input type="reset" name="Submit2" value="Reset" /></td>

</tr>

</table>

</form>

<div id="footer"> copyrights & designedby@Tours & travels</div>

**LOGIN PAGE**

<?php

include("dbconnect.php");

extract($\_POST);

session\_start();

if(isset($\_POST['btn']))

{

$qry=mysqli\_query($conn,"select \* from register where uname='$uname' && psw='$password'");

$num=mysqli\_num\_rows($qry);

if($num==1)

{

$qry1=mysqli\_query($conn,"select \* from register where uname='$uname' && psw='$password'");

$row=mysqli\_fetch\_assoc($qry1);

$\_SESSION['id']=$row['id'];

?>

<script>alert('welcome to User home page');

</script>

<?php

header("location:../userhome.php");

}

else

{

echo "<script>alert('User Name Password Wrong.....')</script>";

}

}

?>

<html>

<title>Tours & Travels</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

text-color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:150px;

background:url("images/4.png");

background-repeat: no-repeat; background-size: 100% 200px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<div id="navbar"><p>Tours And Travels </p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="index.php">Home Page</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="admin.php">Admin login</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a class="active" href="user.php">User Login</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="register.php">User Registration</a></li>

</ul>

<div id="bg1"> </div>

<form id="form1" name="form1" method="post" action="">

<table width="46%" border="0" align="center">

<tr>

<td colspan="2" rowspan="1"><div align="center" class="style1"><strong><font size="+1">User Login</font> </div></td>

</tr>

<tr>

<td width="48%">&nbsp;</td>

<td width="52%">&nbsp;</td>

</tr>

</tr>

<tr>

<td height="31"align="center"><span class="style2"><strong>User Name </strong></span></td>

<td><label>

<input name="uname" type="text" id="uname" />

</label></td>

</tr>

<tr>

<td height="44" align="center"><span class="style2"><strong>Password</strong></span></td>

<td><label>

<input name="password" type="password" id="password" />

</label></td>

</tr>

<tr>

<td>&nbsp;</td>

<td rowspan="2"><label>

<input name="btn" type="submit" id="btn" value="Login" />

<input type="reset" name="Submit2" value="Cancel" />

</label></td>

</tr>

</table>

</form>

<div> &nbsp;</div>

<div id="footer"> copyrights & designedby@Tours & travels</div>

**VIEW BOOKING**

<?php

include("dbconnect.php");

extract($\_POST);

session\_start();

$uid=$\_SESSION['id'];

?>

<html>

<title>Tours & Travels</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

text-color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:150px;

background:url("images/4.png");

background-repeat: no-repeat; background-size: 100% 200px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<div id="navbar"><p>Tours And Travels </p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="userhome.php">UserHome</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="viewt.php">View Transports</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a class="active" href="viewbook.php">View Booking</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="index.php">LogOut</a></li>

</ul>

<div id="bg1"> </div>

<table width="96%" align="center">

<tr>

<td colspan="10" align="center">Packages Booking</td>

</tr>

<tr>

<td width="0%">&nbsp;</td>

<td width="11%"><div align="center" class="style6"><strong>Package Name</strong> </div></td>

<td width="12%"><div align="center" class="style6"><strong>Places</strong> </div></td>

<td width="8%"><div align="center" class="style6"><strong>GST</strong> </div></td>

<td width="11%"><div align="center" class="style6"><strong>End Date</strong> </div></td>

<td width="13%"><div align="center" class="style6"><strong>Pacakge Price</strong> </div></td>

<td width="16%"><div align="center" class="style6"><strong>Number Of persons</strong> </div></td>

<td width="12%"><div align="center" class="style6"><strong>Amount</strong> </div></td>

<td width="15%"><div align="center" class="style6"><strong>Status</strong> </div></td>

</tr>

</form>

<tr>

<td colspan="10">&nbsp;</td>

</tr>

<?php

$qry=mysqli\_query($conn,"select \* from booking where uid='$uid'");

$i=1;

while($row=mysqli\_fetch\_array($qry))

{

$pid=$row['pid'];

$qry1=mysqli\_query($conn,"select \* from package where id='$pid'");

$row1=mysqli\_fetch\_array($qry1);

?>

<tr>

<td width="0%">&nbsp;</td>

<td><div align="center"><?php echo $row1['pname'];?></div></td>

<td><div align="center"><?php echo $row1['place'];?></div></td>

<td><div align="center">5%</div></td>

<td><div align="center"><?php echo $row1['edate'];?></div></td>

<td><div align="center"><?php echo $row1['package\_price'];?></div></td>

<td><div align="center"><?php echo $row['np'];?></div></td>

<td><div align="center"><?php echo $row['amnt'];?></div></td>

<td><div align="center"><?php echo "booked"?></div></td>

<td width="2%"> </div></td>

</tr>

<tr>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<?php

$i++;

}

?>

<tr>

<td colspan="10" align="center">&nbsp;</td>

</tr>

</table>

<table width="96%" align="center">

<tr>

<td colspan="10" align="center">Vehicle Booking</td>

</tr>

<tr>

<td width="1%">&nbsp;</td>

<td width="12%"><div align="center" class="style6"><strong>Vehicle Name</strong> </div></td>

<td width="20%"><div align="center" class="style6"><strong>No Of Seats</strong> </div></td>

<td width="15%"><div align="center" class="style6"><strong>Vehicle Number</strong> </div></td>

<td width="12%"><div align="center" class="style6"><strong>Amount Per Day</strong> </div></td>

<td width="11%"><div align="center" class="style6"><strong>Start Date</strong> </div></td>

<td width="15%"><div align="center" class="style6"><strong>End Date</strong> </div></td>

<td width="15%"><div align="center" class="style6"><strong>Total Amount</strong> </div></td>

<td width="12%"><div align="center" class="style6"><strong>Status</strong> </div></td>

</tr>

</form>

<tr>

<td colspan="10">&nbsp;</td>

</tr>

<?php

$qry=mysqli\_query($conn,"select \* from book where uid='$uid'");

$i=1;

while($row=mysqli\_fetch\_array($qry))

{

$gid=$row['rid'];

$qry1=mysqli\_query($conn,"select \* from transport where id='$gid'");

$row1=mysqli\_fetch\_array($qry1);

?>

<tr>

<td width="1%">&nbsp;</td>

<td><div align="center"><?php echo $row1['vname'];?></div></td>

<td><div align="center"><?php echo $row1['seats'];?></div></td>

<td><div align="center"><?php echo $row1['vno'];?></div></td>

<td><div align="center"><?php echo $row1['amnt'];?></div></td>

<td><div align="center"><?php echo $row['sdate'];?></div></td>

<td><div align="center"><?php echo $row['edate'];?></div></td>

<td><div align="center"><?php echo $row['tamnt'];?></div></td>

<td><div align="center"><?php echo "booked"?></div></td>

<td width="2%"> </div></td>

</tr>

<tr>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<?php

$i++;

}

?>

<tr>

<td colspan="10" align="center">&nbsp;</td>

</tr>

</table>

<div> &nbsp;</div>

<div id="footer"> copyrights & designedby@Tours & travels</div>

**ADMIN DASHBOARD**

<?php

include("dbconnect.php");

extract($\_POST);

session\_start();

if(isset($\_POST['btn']))

{

$file\_name=$\_FILES['img']['name'];

$file\_loc=$\_FILES['img']['tmp\_name'];

$folder = "upload/";

$path=move\_uploaded\_file($file\_loc,$folder.$file\_name);

$img=$file\_name;

$qry=mysqli\_query($conn,"insert into package(pname,package\_price,place,sdate,edate,img,des)values('$pname','$pp','$place','','$days','$img','$des')");

if($qry)

{

echo "<script>alert('Data Save');</script>";

}

else

{

echo "<script>alert('Data Not Save');</script>";

}

}

?>

<html>

<title>Tours & Travels</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:150px;

background:url("images/4.png");

background-repeat: no-repeat; background-size: 100% 200px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<div id="navbar"><p>Tours And Travels </p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a class="active" href="adminhome.php">Home Page</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="transopts.php">Add Transopts</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="viewu.php">User Details</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="viewp.php">Packages Details</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="booking.php">Booking Details</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="index.php">LogOut</a></li>

</ul>

<div id="bg1"> </div>

<form id="form1" name="form1" method="post" action="" enctype="multipart/form-data">

<table width="46%" border="0" align="center">

<tr>

<td colspan="2" rowspan="1"><div align="center" class="style1"><strong><font size="+1">Add Packages</font> </div></td>

</tr>

<tr>

<td width="48%">&nbsp;</td>

<td width="52%">&nbsp;</td>

</tr>

</tr>

<tr>

<td height="31"align="center"><span class="style2"><strong>Package Name </strong></span></td>

<td><label>

<input name="pname" type="text" id="pname" />

</label></td>

</tr>

<tr>

<td height="44" align="center"><span class="style2"><strong>Package Price </strong></span></td>

<td><label>

<input name="pp" type="text" id="pp" />

</label></td>

</tr>

<tr>

<td height="44" align="center"><span class="style2"><strong>Places</strong></span></td>

<td><label>

<textarea name="place" type="text" ></textarea>

</label></td>

</tr>

<tr>

<td height="44" align="center"><span class="style2"><strong>Number Of Days</strong></span></td>

<td><label>

<input name="days" type="text" id="days" />

</label></td>

</tr>

<tr>

<td height="44" align="center"><span class="style2"><strong>Images</strong></span></td>

<td><label>

<input name="img" type="file" id="img" />

</label></td>

</tr>

<tr>

<td height="44" align="center"><span class="style2"><strong>Description</strong></span></td>

<td><label>

<textarea name="des" id="des" ></textarea>

</label></td>

</tr>

<tr>

<td>&nbsp;</td>

<td rowspan="2"><label>

<input name="btn" type="submit" id="btn" value="add" />

<input type="reset" name="Submit2" value="Cancel" />

</label></td>

</tr>

</table>

</form>

<div> &nbsp;</div>

<div id="footer"> copyrights & designedby@Tours & travels</div>

**PAYMENT**

<?php

include("dbconnect.php");

extract($\_POST);

session\_start();

$uid=$\_SESSION['id'];

$tamnt=$\_REQUEST['tmant'];

if(isset($\_POST['btn']))

{

?>

<script language="javascript">

alert("Amount Transfer Successfull..");

window.location.href="userhome.php";

</script>

<?php

}

?>

<script>

function cno()

{

var cno=/^[0-9]{16}$/;

if(document.f1.cno.value.search(cno)==-1)

{

alert("enter correct Card no");

document.f1.cno.focus();

return false;

}

}

function cvv()

{

var cvv=/^[0-9]{3}$/;

if(document.f1.cvv.value.search(cvv)==-1)

{

alert("enter correct Pin");

document.f1.cvv.focus();

return false;

}

}

function vali()

{

var cno=/^[0-9]{16}$/;

var cvv=/^[0-9]{3}$/;

//var mesg=/^[a-zA\_Z0-9]{10,300}$/;

if(document.f1.cno.value.search(cno)==-1)

{

alert("the card number Should be 16 degit");

document.f1.cno.focus();

return false;

}

else if(document.f1.cvv.value.search(cvv)==-1)

{

alert("The Cvv number should be 3 degit");

document.f1.cvv.focus();

return false;

}

else

{

return true;

}

}

</script>

<html>

<title>Tours & Travels</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:150px;

background:url("images/4.png");

background-repeat: no-repeat; background-size: 100% 200px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<div id="navbar"><p>Tours And Travels </p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a class="active" href="userhome.php">UserHome</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="viewt.php">View Transports</a></li> <li><a href="#">&nbsp;</a></li>

<li><a href="viewbook.php">View Booking</a></li> <li><a href="#">&nbsp;</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="index.php">LogOut</a></li>

</ul>

<div id="bg1"> </div>

<div> &nbsp;</div>

<form method="post" id="f1" name="f1" action="#" onSubmit="return vali()">

<table width="50%" border="0" align="center">

<tr>

<td colspan="2"><span class="style1">Payment Mode.... </span></td>

</tr>

<tr>

<td>&nbsp;</td>

<td><label>

<input type="image" name="imageField" src="images/payment.png" />

</label></td>

</tr>

<tr>

<td width="35%"><span class="style4">Amount</span> </td>

<td> <?php echo $tamnt;?></td>

</tr>

<tr>

<td height="33"><span class="style4">Enter Card Number </span></td>

<td><input name="cno" type="text" id="cno" /></td>

</tr>

<tr>

<td height="36"><span class="style4">CVV Number </span></td>

<td><input name="cvv" type="password" id="cvv" /></td>

</tr>

<tr>

<td><span class="style4">Card Name </span></td>

<td><input name="cname" type="text" id="cname" required /></td>

</tr>

<tr>

<td>&nbsp;</td>

<td><input name="btn" type="submit"value="Pay" /></td>

</tr>

</table>

</form>

<br>

<br>

<br>

<br>

<br>

<br>

<br />

<br />

<br />

<br />

<br>

<br />

<br />

<br />

<div> &nbsp;</div>

<div id="footer">Designed By Admin</div>

**USER HOME**

<?php

include("dbconnect.php");

extract($\_POST);

session\_start();

$uid=$\_SESSION['id'];

?>

<html>

<title>Tours & Travels</title>

<style>

#navbar {

padding: 25px;

background:#00ffff;

background-size: 1420px 100px;

text-align:center;

text-decoration:blink;

text-color:#4d4d00;

font-family: Arial;

font-size:35px;

}

p

{

color:#4d4d00;

text-align: center;

text-transform: uppercase;

font-size:20px;

}

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #888844;

position: -webkit-sticky; /\* Safari \*/

position: sticky;

top: 0;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover {

background-color: #111;

}

.active {

background-color: #4CAF50;

}

#footer {

border: 2px solid #888844;

padding: 45px;

background: #888844;

background-repeat: no-repeat;

background-size: 1420px 100px;

border-radius:10px;

text-align:center;

text-decoration:blink;

font-family: Arial;

font-size:15px;

}

#bg1 {

padding:150px;

background:url("images/4.png");

background-repeat: no-repeat; background-size: 100% 200px;

border-radius:5px;

border-radius:10px;

font-size:35px;

}

</style>

</head>

<div id="navbar"><p>Tours And Travels </p></div>

<ul>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a href="#">&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;</a></li>

<li><a class="active" href="userhome.php">UserHome</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="viewt.php">View Transports</a></li> <li><a href="#">&nbsp;</a></li>

<li><a href="viewbook.php">View Booking</a></li> <li><a href="#">&nbsp;</a></li>

<li><a href="#">&nbsp;</a></li>

<li><a href="index.php">LogOut</a></li>

</ul>

<div id="bg1"> </div>

<table width="96%" align="center">

<tr>

<td colspan="10">&nbsp;</td>

</tr>

<tr>

<td width="0%">&nbsp;</td>

<td width="5%"><div align="center" class="style6"><strong>Id</strong> </div></td>

<td width="10%"><div align="center" class="style6"><strong>Package Name</strong> </div></td>

<td width="11%"><div align="center" class="style6"><strong>Places</strong> </div></td>

<td width="9%"><div align="center" class="style6"><strong>GST</strong> </div></td>

<td width="9%"><div align="center" class="style6"><strong>Number of days</strong> </div></td>

<td width="14%"><div align="center" class="style6"><strong>Amount(per one person)</strong> </div></td>

<td width="16%"><div align="center" class="style6"><strong>Image</strong> </div></td>

<td width="21%"><div align="center" class="style6"><strong>Description</strong> </div></td>

<td width="21%"><div align="center" class="style6"><strong>Book</strong> </div></td>

</tr>

</form>

<tr>

<td colspan="10">&nbsp;</td>

</tr>

<?php

$qry=mysqli\_query($conn,"select \* from package");

$i=1;

while($row=mysqli\_fetch\_array($qry))

{

?>

<tr>

<td width="0%">&nbsp;</td>

<td><div align="center"><?php echo $row['id'];?></div></td>

<td><div align="center"><?php echo $row['pname'];?></div></td>

<td><div align="center"><?php echo $row['place'];?></div></td>

<td><div align="center">5%</div></td>

<td><div align="center"><?php echo $row['edate'];?></div></td>

<td><div align="center"><?php echo $row['package\_price'];?></div></td>

<td><div align="center"><img src="upload\<?php echo $row['img'];?>"width="100"height="100"></div></td>

<td><div align="center"><?php echo $row['des'];?></div></td>

<td width="5%"><div align="center"><a href="book.php?pid=<?php echo $row['id'];?>&pp=<?php echo $row['package\_price'];?>">Book Pakages</a></div></td>

</tr>

<tr>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

<td>&nbsp;</td>

</tr>

<?php

$i++;

}

?>

<tr>

<td colspan="10" align="center">&nbsp;</td>

</tr>

</table>

<div> &nbsp;</div>

<div id="footer"> copyrights & designedby@Tours & travels</div>

1. **SCREEN SHOTS**