# Developer Guide: Travel Website (PHP & MySQL)

## Table of Contents

1. \*\*Introduction\*\*

- 1.1 Project Overview

- 1.2 Technologies Used

2. \*\*Setting Up the Development Environment\*\*

- 2.1 Server Configuration

- 2.2 Database Setup

- 2.3 Codebase Setup

5. \*\*PHP Coding Standards\*\*

- 5.1 Naming Conventions

- 5.2 Code Organization

- 5.3 Security Best Practices

6. \*\*User Authentication\*\*

- 6.1 User Registration

- 6.2 User Login

- 6.3 User Sessions

7. \*\*Travel Plans Module\*\*

- 7.1 Displaying Travel Plans

- 7.2 Plan Details Page

8. \*\*Inquiry Module\*\*

- 8.1 Inquiry Form

- 8.2 Processing Inquiries

- 8.3 Admin Notifications

9. \*\*Admin Panel\*\*

- 9.1 Admin Login

- 9.2 Managing User Inquiries

- 9.3 Responding to Inquiries

10. \*\*Security Considerations\*\*

- 10.1 Input Validation

- 10.2 SQL Injection Prevention

- 10.3 Cross-Site Scripting (XSS) Prevention

11. \*\*Testing\*\*

- 11.1 Unit Testing

- 11.2 User Acceptance Testing

12. \*\*Deployment\*\*

- 12.1 Server Deployment

- 12.2 Database Deployment

- 12.3 Configuring Environment Variables

13. \*\*Maintenance and Updates\*\*

- 13.1 Version Control

- 13.2 Updating Dependencies

- 13.3 Monitoring and Logging

---

## 1. Introduction

### 1.1 Project Overview

The Travel Website is a PHP and MySQL-based web application that allows users to explore travel plans, submit inquiries, and receive responses from the admin. This developer guide provides insights into the project structure, key functionalities, and best practices for development.

### 1.2 Technologies Used

- PHP

- MySQL

- HTML, CSS

- JavaScript (if applicable)

- LAMP Stack (Linux, Apache, MySQL, PHP)

## 2. Setting Up the Development Environment

### 2.1 Server Configuration

Ensure your development environment is configured with a compatible version of Apache, PHP, and MySQL. Use tools like XAMPP or WampServer for a quick setup.

### 2.2 Database Setup

Create a MySQL database for the project and configure the database connection parameters in the project's configuration files.

### 2.3 Codebase Setup

Clone the project repository to your local machine and set up the necessary dependencies using Composer (if applicable). Ensure your project follows PSR (PHP-FIG) coding standards.

## 5. PHP Coding Standards

### 5.1 Naming Conventions

Follow PSR-1 and PSR-2 standards for class and method naming.

### 5.2 Code Organization

Organize code into classes and use namespaces for modular and maintainable code.

### 5.3 Security Best Practices

- Use prepared statements to prevent SQL injection.

- Sanitize user inputs to prevent cross-site scripting (XSS) attacks.

- Implement password hashing for user authentication.

## 6. User Authentication

## 7. Travel Plans Module

### 7.1 Displaying Travel Plans

Fetch travel plans from the database and display them on the website.

### 7.2 Plan Details Page

Create a detailed view for each travel plan, including itinerary, cost, and other relevant information.

## 8. Inquiry Module

### 8.1 Inquiry Form

Build a form for users to submit inquiries for specific travel plans.

### 8.2 Processing Inquiries

Handle form submissions, validate data, and store inquiries in the database.

### 8.3 Admin Notifications

Implement a notification system to alert admin about new inquiries.

## 9. Admin Panel

### 9.1 Admin Login

Create a secure login system for admin access to the admin panel.

### 9.2 Managing User Inquiries

Allow admins to view and manage user inquiries, including responding to inquiries.

### 9.3 Responding to Inquiries

Implement a system for admins to respond to user inquiries directly from the admin panel.

## 10. Security Considerations

### 10.1 Input Validation

Validate all user inputs to prevent malicious data entry.

### 10.2 SQL Injection Prevention

Use prepared statements and parameterized queries to prevent SQL injection attacks.

### 10.3 Cross-Site Scripting (XSS) Prevention

Sanitize user inputs before rendering them to prevent XSS attacks.

## 11. Testing

### 11.1 Unit Testing

Write unit tests for critical functions and classes to ensure functionality.

### 11.2 User Acceptance Testing

Conduct thorough testing of user workflows to identify and fix any issues.

## 12. Deployment

### 12.1 Server Deployment

Deploy the application

to a production server following best practices.

### 12.2 Database Deployment

Deploy the database schema and initial data to the production database.

### 12.3 Configuring Environment Variables

Use environment variables for configuration settings, such as database credentials.

## 13. Maintenance and Updates

### 13.1 Version Control

Use version control (e.g., Git) for tracking changes and collaborating with other developers.

### 13.2 Updating Dependencies

Regularly update dependencies and libraries to benefit from security patches and new features.

### 13.3 Monitoring and Logging

Implement monitoring tools and logging mechanisms to track errors and user interactions for troubleshooting.

---

This Developer Guide provides an overview of the Travel Website project's structure, key functionalities, and best practices for development. Follow the guidelines to contribute to the project or customize it according to specific requirements.