Software Design Document (SDD)

Project: MakeYourTrip Web Portal

Prepared for: [Your Client / Company Name]

Prepared by: [Your Name]

Date: [Insert Date]

1. Introduction

1.1 Purpose

This document provides the detailed software design for the MakeYourTrip Web Portal. It includes architectural and component-level designs to guide implementation and development.

1.2 Scope

The system includes a customer-facing portal and a tour operator extranet. It facilitates tour creation, listing, booking, payments, cancellations, and personalized operator branding.

1.3 Definitions

- UI: User Interface
- API: Application Programming Interface
- DBMS: Database Management System
- MVC: Model View Controller

2. System Overview

The system follows a multi-tier architecture comprising a presentation layer, application logic layer, and a data storage layer. It is designed for modularity and scalability.

3. System Architecture

3.1 High-Level Architecture

The system is divided into:

- Frontend: Web interfaces for customers and operators.
- Backend: RESTful API services.
- Database: Stores user data, tour data, booking history, etc.
- Third-party integrations: Google Maps, Payment Gateways, PDF generation.

3.2 Component Diagram Description

Main components include:
- Authentication Module
- Tour Management Module
- Booking Engine
- Payment Gateway Integration
- Notification System
- Admin Dashboard

4. Component Design

4.1 Customer Portal

Allows users to search, compare, book, view history, cancel, and download tour information. Includes authentication, booking, and payment modules.

4.2 Operator Extranet

Allows tour operators to create, manage, and publish tour packages. Includes modules for profile customization, multimedia management, and itinerary building.

4.3 Admin Panel

Monitors system activities, manages users, and handles disputes or edge-case bookings.

5. Data Design

5.1 Database Schema

Core entities:
- Users (Customer, Operator, Admin)
- Tours
- Bookings
- Payments
- Locations
- Vehicles
- Accommodations

5.2 ER Diagram Description

The schema is normalized. Relationships are:
- One-to-many: Operator to Tours
- Many-to-many: Tour to Themes/Destinations
- One-to-one: Booking to Payment

6. Interface Design

6.1 User Interface

Built using responsive web technologies (HTML5, CSS3, JavaScript, React or Angular). Designed for ease of navigation and accessibility.

6.2 API Design

RESTful APIs with endpoints for:
- Authentication
- Booking
- Payments
- Tour search and filters
- Notifications

7. Security Design

Implements role-based access control, encrypted passwords, secure API endpoints (HTTPS), token-based authentication (JWT), and data validation to prevent injection attacks.

8. Performance Considerations

Optimized database queries, load balancing, caching for tour data, and asynchronous processing for email/PDF generation.