Information Systems and Data Modeling – IT1090

Assignment

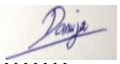
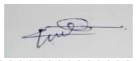
Title: Hotel reservation system for tourists	
Batch Number: Y1.S2.WD.06.01	Group Number: MLB_06.01_03

Declaration:

We hold a copy of this assignment that we can produce if the original is lost or damaged.

We hereby certify that no part of this assignment has been copied from any other group's work or from any other source. No part of this assignment has been written / produced for our group by another person except where such collaboration has been authorized by the subject lecturer/tutor concerned.

Group Members:

IT23257504	Harindu Weligepola	 ..... signature
IT23265974	Danuja Liyanage	 ..... signature
IT23268180	Dilshan Mahavithana	 ..... signature
IT23256446	Mohammed Ruhaim	 ..... signature
IT23257436	Wishwa Ekanayake	 ..... signature

Submitted on: <02/04/2024>

Table of Contents

Table of Contents

Terms and References

The basis for this report is the "Hotel Reservation Management System," presented to meet the requirements of the Sri Lanka Institute of Information Technology's Information System and Data Modeling (ISDM)-IT1090 module. The Sri Lanka Institute of Information Technology's Information System and Data Modeling (ISDM)-IT1090 module criteria.

Acknowledgment

This project would not have been able to be finished without the help and direction of several people. The module's lead instructors and the other assistant lecturers, for their thoughtful advice and support with the work.

The videos that were posted on the course website improved our understanding of the proper methods for conducting research and completing the final project. We also want to express our gratitude to the professors who took the time to answer our emails.

We also thank our colleagues who helped us out by offering their ideas and tips on how to make the report better. The work and assistance would have been necessary for the task to be finished.

Introduction

In crafting our hotel reservation system for tourists, we delved into the realm of online connectivity between businesses and customers, recognizing the paramount convenience it offers to clients. Consequently, we devised a solution tailored to the hospitality industry, specifically focusing on hotel reservations. Our endeavor led to the creation of a user-friendly website where individuals can seamlessly browse and book hotel accommodation online. Through our platform, users can effortlessly register, access detailed information about available services, receive customer support, and provide valuable feedback.

The development of a hotel reservation system necessitates a thorough understanding of various technologies. Key among these are programming languages such as HTML, CSS, and JavaScript, which enable the creation of dynamic and interactive web pages. Additionally, the utilization of relational databases, exemplified by MySQL, plays a crucial role in storing and managing data efficiently. By centralizing data storage on a computer, our system streamlines data management processes, thereby saving time and effort for both customers and hotel staff. Moreover, the inherent security features of database storage mitigate the risk of data loss, ensuring that users retain control over their information while benefiting from reliable and uninterrupted security services.

Hypothetical Scenario

In our hotel reservation system tailored for tourists, guests interested in booking hotel accommodations can easily access our website and peruse the available packages. Upon arrival, they have the option to register as customers within the system. To initiate registration, guests must complete a registration form, furnishing their contact details such as name, age, address, telephone number, and email. Additionally, they are required to create a username and password. Upon submission, the system triggers a verification email to confirm the account. Once the verification process is complete, the account is successfully created.

Registered users can then log in to the website and explore the array of hotel reservation packages available. System administrators maintain access to update package availability and ensure their accuracy. Users can proceed to book their desired package and make payments either online or offline. Subsequently, the system generates a unique payment ID, subject to approval by the manager. Upon payment verification, customers receive an email confirming successful payment processing.

Following their hotel stay, users are invited to provide feedback on their experience. The system's support agents manage customer feedback, ensuring a seamless process. Additionally, users encountering any difficulties during the registration or booking process can avail themselves of customer support services through inquiries handled by dedicated customer support agents.

Requirement Analysis

Functional Requirements

Tourist, system administrator, manager, marketing manager, and hotel provider can access the system. While tourist can only access the front end of the system other users can access the back end of the system.

1. **Tourist**

   User Requirements

   - Tourists should be able to browse available hotel accommodation packages.
   - They should be able to register as new users and create an account.
   - Tourists must fill out a registration form with personal information and accept terms.
   - They should have the ability to create a unique username and password.
   - Tourists should complete a verification process for account activation.
   - Once registered, tourists can log in to the system.
   - They should be able to view details of hotel packages, including amenities, prices, and availability.
   - Tourists can book hotel accommodations online.
   - They should be able to make payments for bookings, either online or offline.
   - After their stay, tourists can leave feedback on their experience.

   System Requirements

   - The system must provide a user-friendly interface for tourists.
   - Online registration should be secure and include mechanisms for verification.
   - Real-time validation checks should be implemented during the registration process.

- The system should require tourists to log in before accessing booking functionalities.
- Error handling mechanisms should prompt users to re-enter login details if authentication fails.
- User accounts must be updated accurately after each interaction, including bookings and feedback submissions.

2. **System Administrator**

User Requirements

- System administrators should have privileged access to all functionalities of the system.
- They must be able to log in to the system using administrative credentials.
- Administrators should have the ability to manage user accounts, including verification, editing, and deletion.

System Requirements

- A dedicated dashboard is needed for system administrators to manage user accounts efficiently.
- The administrator should be able to verify new user accounts and edit existing accounts as necessary.
- An option to delete user accounts should be available for system administrators.
- The system should allow administrators to generate various reports for analysis and decision-making purposes.

3. **Manager**

User Requirements

- Managers should be able to log in to the system using their credentials.
- They must have the ability to update the availability of hotel packages.
- Managers should be able to verify payments made by tourists.

System Requirements

- A dashboard is required for managers to view and manage hotel accommodation packages.
- Managers should be able to add new hotel packages and update existing ones as needed.
- Payment verification functionality should be available for managers to ensure financial transactions are processed accurately.
- The system should provide options for generating various types of reports relevant to hotel package management.

4. **Marketing Manager**

User Requirements

- Marketing managers should be able to log in to the system.
- They must have access to view and analyse data related to hotel package bookings and feedback.
- Marketing managers should be able to create and manage promotional campaigns.

System Requirements

- A dashboard is required for marketing managers to access relevant data and analytics.
- The system should provide tools for creating and managing promotional campaigns, such as discounts and special offers.
- Access to comprehensive reports on booking trends, customer feedback, and campaign effectiveness should be available for marketing managers.

5. **Hotel Provider:**

User Requirements

- Hotel providers should be able to log in to the system using their credentials.

- They must have the ability to manage hotel information, including room availability, rates, and descriptions.
- Hotel providers should be able to update their hotel's availability and details in real-time.

System Requirements

- A dedicated dashboard is needed for hotel providers to manage their hotel information efficiently.
- The system should allow hotel providers to update room availability, rates, and descriptions easily.
- Real-time synchronization between the system and hotel provider databases is necessary to ensure accurate and up-to-date information for tourists browsing and booking hotel accommodations.

Non-Functional Requirements

1. Speed

    - The system should be able to quickly send and retrieve data, and the system should be able to work fast.

2. Size

    - System processing power should not be overused, and the required parts must be loaded first.

3. Ease of use

    - The system should be easy to navigate and use for both customers and staff.

4. Security

    - The system should be secure and protect customer data from unauthorized access.

5. Maintainability

    - The system should be easy to maintain and update.

6. Performance

    - The system should be able to handle a large number of concurrent users and transaction without any performance degradation.

Data Requirements of the System

1. Tourist

    - User ID (UserID)
    - Name
    - Username
    - Address
    - Password
    - Email
    - Age
    - Phone Number (PhoneNo)

2. Manager

    - Manager ID (ManagerID)
    - Phone Number (PhoneNo)
    - Name
    - Email

3. System Administrator

    - Admin ID (AdminID)
    - Name
    - Email

4. Marketing manager

    - Marketing Manager ID (MManagerID)
    - Name
    - Email
    - Phone Number (PhoneNo)

5. Hotel Provider

    - Provider ID (ProviderID)
    - Name
    - Email
    - Phone Number (PhoneNo)

6. Package

    - Package ID (PackageID)
    - Package Name (PackageName)
    - Description
    - Availability
    - Price
    - Provider ID (ProviderID)
    - Marketing Manager ID (MManagerID)

7. Booking

    - Booking ID (BookingID)
    - Package ID (PackageID)
    - User ID (UserID)
    - Booking Date (BookingDate)
    - Payment Status (PaymentStatus)

8. Payment

    - Payment ID (PaymentID)
    - Booking ID (BookingID)
    - Payment Amount (PaymentAmount)
    - Payment Method (PaymentMethod)
    - Payment Date (PaymentDate)

9. Feedback

    - Feedback ID (FeedbackID)
    - User ID (UserID)
    - Feedback Text (FeedbackText)
    - Feedback Date (FeedbackDate)

10. Support Inquiry

    - Inquiry ID (InquiryID)
    - User ID (UserID)
    - Inquiry Text (InquiryText)
    - Inquiry Date (InquiryDate)
    - Inquiry Status (InquiryStatus)

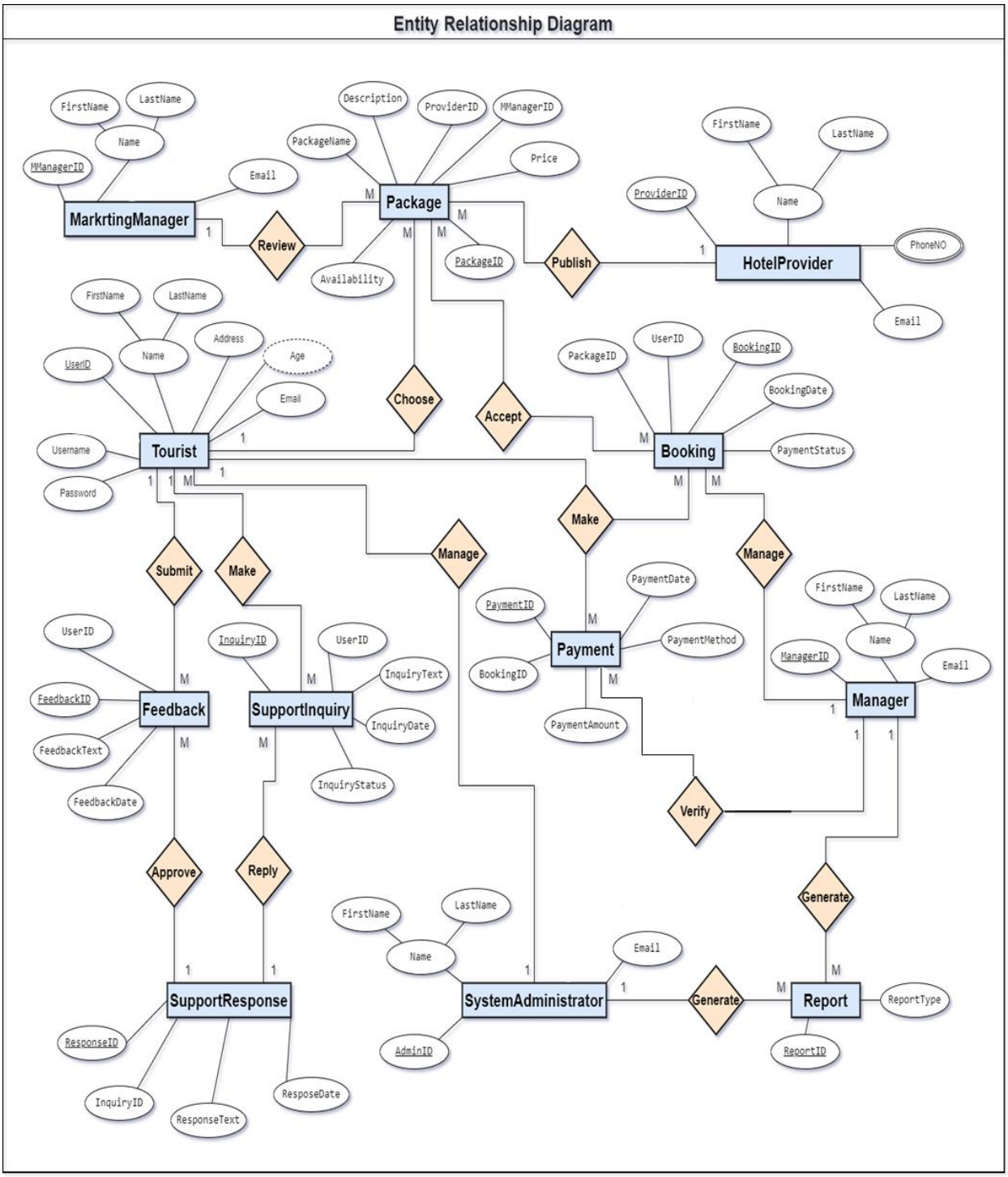
11. Support Response

    - Response ID (ResponseID)
    - Inquiry ID (InquiryID)
    - Response Text (ResponseText)
    - response Date (ResposeDate)

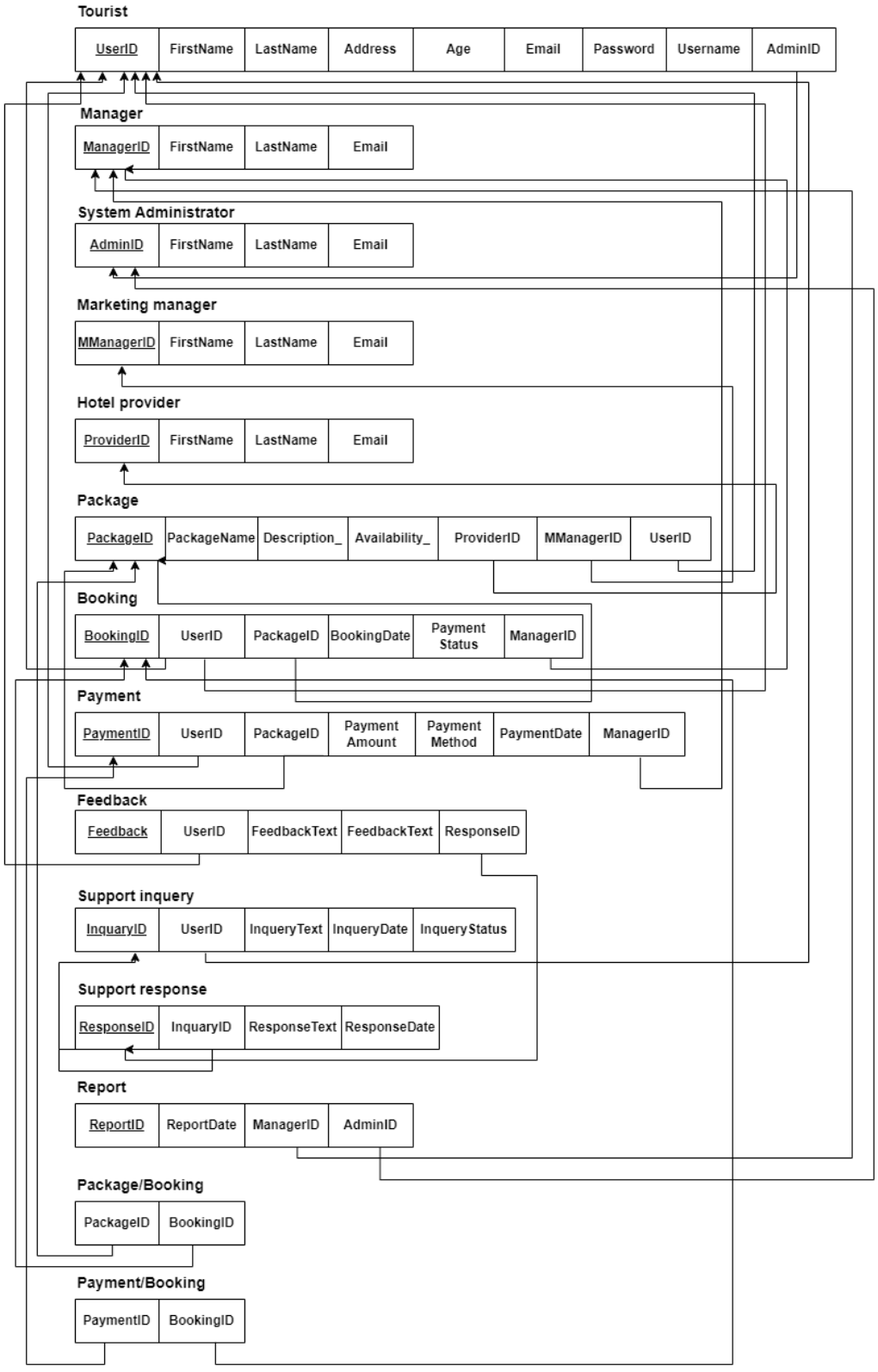
12. Report

    - Report ID (ReportID)
    - Report Type (ReportType)

Entity Relationship Diagram



Relational Schema



SQL Queries to create a Data Base

```
--------Hotel reservation Management System --------


--------Administrator Table --------

CREATE TABLE Administrator (
    AdminID INT PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL
);

INSERT INTO Administrator (AdminID, FirstName, LastName, Email)
VALUES ('001', 'Peterson', 'Andrew', 'petersonandrew@gmail.com');

SELECT * FROM Administrator;




--------Hotel Provider Table --------

CREATE TABLE Hotel_Provider (
    ProviderID INT PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL
);

INSERT INTO Hotel_Provider (ProviderID, FirstName, LastName, Email)
VALUES
    ('101', 'John', 'Doe', 'johndoe@example.com'),
    ('102', 'Alice', 'Smith', 'alice.smith@example.com'),
    ('103', 'Michael', 'Johnson', 'michael.johnson@example.com'),
    ('104', 'Emily', 'Brown', 'emily.brown@example.com'),
    ('105', 'David', 'Martinez', 'david.martinez@example.com');

SELECT * FROM Hotel_Provider;




--------Manager Table --------

CREATE TABLE Manager (
    ManagerID INT PRIMARY KEY NOT NULL,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL
);
```

```
INSERT INTO Manager (ManagerID, FirstName, LastName, Email)
VALUES ('101', 'David', 'Miller', 'davidmiller@gmail.com');

SELECT * FROM Manager;



--------Marketing Manager Table --------

CREATE TABLE MarketingManager (
    MManagerID INT PRIMARY KEY NOT NULL,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL
);

INSERT INTO MarketingManager (MManagerID, FirstName, LastName, Email)
VALUES ('201', 'Calvin', 'Klein', 'calvinklein@gmail.com');

SELECT * FROM MarketingManager;



--------Package Table --------
CREATE TABLE Package_ (
    PackageID INT NOT NULL,
    PackageName VARCHAR(255) NOT NULL,
    Description_ VARCHAR(255) NOT NULL,
    Availability_ VARCHAR(255) NOT NULL,
    ProviderID INT NOT NULL,
    ManagerID INT NOT NULL,
    UserID INT NOT NULL,
    CONSTRAINT Package__PK PRIMARY KEY (PackageID),
    CONSTRAINT packagemanagerFK FOREIGN KEY (ManagerID) REFERENCES Manager (ManagerID),
    CONSTRAINT packagetouristFK FOREIGN KEY (UserID) REFERENCES Tourist (UserID),
    CONSTRAINT packackageproviderFK FOREIGN KEY (ProviderID) REFERENCES Hotel_Provider
(ProviderID)
);

INSERT INTO Package_ (PackageID, PackageName, Description_, Availability_, ProviderID,
ManagerID, UserID)
VALUES
    ('01', '2 person', '2 persons per day with a double bed 12h', 'Available', '101',
'101', '001'),
    ('02', '3 person', '3 persons per day with 3 single beds 12h', 'Available', '102',
'101', '002'),
    ('03', 'One day', 'Single room for one day 24h', 'Not Available', '103', '101',
'003'),
    ('04', 'One day 2 person', 'Single room for one day 24h with a double bed',
'Available', '104', '101', '004'),
    ('05', 'Air conditioned', 'Air conditioned room with a king-size bed', 'Not
Available', '105', '101', '005');


SELECT * FROM Package_;
```

```
--------Tourist Table --------

CREATE TABLE Tourist (
    UserID INT NOT NULL,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Address VARCHAR(255) NOT NULL,
    Age INT NOT NULL,
    Email VARCHAR(255) NOT NULL,
    Username VARCHAR(255) NOT NULL,
    Password VARCHAR(255) NOT NULL,
    AdminID INT NOT NULL,
    CONSTRAINT TouristPK PRIMARY KEY (UserID),
    CONSTRAINT TouristAdminFK FOREIGN KEY (AdminID) REFERENCES Administrator (AdminID)
);

INSERT INTO Tourist (UserID, FirstName, LastName, Address, Age, Email, Username,
Password, AdminID)
VALUES
    ('001', 'Shane', 'Watson', '1/23 Main Street, Mel', '25', 'shanewatson@gmail.com',
'shane.watson', 'password', '001'),
    ('002', 'Amal', 'Jackson', '456 Kottawa, Colombo', '23', 'amaljackson@gmail.com',
'samal.jackson', 'password', '001'),
    ('003', 'Kamal', 'Wayne', '110 Malabe', '28', 'kamalwayne@gmail.com', 'kamal.wayne',
'password', '001'),
    ('004', 'Peter', 'Perera', '2090 Ratnapura', '26', 'peterperera@gmail.com',
'speter.perera', 'password', '001'),
    ('005', 'Vinuji', 'Fernando', '1/40 Kuruwita', '30', 'vinujifernando@gmail.com',
'vinuji.fernando', 'password', '001');

SELECT * FROM Tourist;



--------Booking Table --------

CREATE TABLE Booking (
    BookingID INT NOT NULL,
    UserID INT NOT NULL,
    PackageID INT NOT NULL,
    BookingDate DATE NOT NULL,
    PaymentStatus VARCHAR(255) NOT NULL,
    ManagerID INT NOT NULL,
    CONSTRAINT bookingPK PRIMARY KEY (BookingID),
    CONSTRAINT bookingmanagerFK FOREIGN KEY (ManagerID) REFERENCES Manager (ManagerID),
    CONSTRAINT bookingtouristFK FOREIGN KEY (UserID) REFERENCES Tourist (UserID),
    CONSTRAINT bookingpackageFK FOREIGN KEY (PackageID) REFERENCES Package_ (PackageID)
);


INSERT INTO Booking (BookingID, UserID, PackageID, BookingDate, PaymentStatus, ManagerID)
VALUES
    ('0001', '001', '01', '2024-05-02', 'Payment successful', '101'),
    ('0002', '002', '02', '2024-05-01', 'Payment successful', '101'),
    ('0003', '003', '03', '2024-04-28', 'Payment successful', '101'),
    ('0004', '004', '04', '2024-05-03', 'Payment successful', '101'),
    ('0005', '005', '05', '2024-05-03', 'Payment successful', '101')
```

```sql
SELECT * FROM Booking;



--------Support_inquery Table --------

CREATE TABLE Support_inquery (
    InquiryID INT NOT NULL,
    UserID INT NOT NULL,
    InqueryText VARCHAR(255) NOT NULL,
    InqueryDate DATE NOT NULL,
    InqueryStatus VARCHAR(255) NOT NULL,
    CONSTRAINT Support_inquery_PK PRIMARY KEY (InquiryID),
    CONSTRAINT Support_inquery_User_FK FOREIGN KEY (UserID) REFERENCES Tourist (UserID)
);

INSERT INTO Support_inquery (InquiryID, UserID, InqueryText, InqueryDate, InqueryStatus)
VALUES
    ('1', '001', 'How can I reset my password?', '2024-05-02', 'Pending'),
    ('2', '002', 'I am having trouble accessing my account.', '2024-05-01', 'Resolved'),
    ('3', '003', 'I need assistance with my recent order.', '2024-04-30', 'Pending'),
    ('4', '004', 'My payment is not going through.', '2024-04-29', 'Open'),
    ('5', '005', 'I would like to change my subscription plan.', '2024-04-28', 'Open');

SELECT * FROM Support_inquery;



--------Support Response Table --------

CREATE TABLE Response_ (
    ResponseID INT NOT NULL,
    UserID INT NOT NULL,
    InquiryID INT NOT NULL,
    ResponseText VARCHAR(255) NOT NULL,
    ResponseDate DATE NOT NULL,
    CONSTRAINT Response_PK PRIMARY KEY (ResponseID),
    CONSTRAINT Response_User_FK FOREIGN KEY (UserID) REFERENCES Tourist (UserID),
    CONSTRAINT Response_Inquiry_FK FOREIGN KEY (InquiryID) REFERENCES Support_inquery
(InquiryID)
);

INSERT INTO Response_ (ResponseID, UserID, InquiryID, ResponseText, ResponseDate)
VALUES
    ('1', '001', '1', 'Your password has been reset successfully.', '2024-05-03'),
    ('2', '002', '2', 'We have resolved the issue with your account access.', '2024-05-
02'),
    ('3', '003', '3', 'Our team is currently working on resolving your order issue.',
'2024-05-01'),
    ('4', '004', '4', 'Please check your payment details and try again.', '2024-04-30'),
    ('5', '005', '5', 'We have updated your subscription plan as requested.', '2024-04-
29');

SELECT * FROM Response_;
```

```
--------Feedback Table --------

CREATE TABLE Feedback_ (
    FeedbackID INT NOT NULL,
    UserID INT NOT NULL,
    FeedbackText VARCHAR(255) NOT NULL,
    FeedbackDate DATE NOT NULL,
    ResponseID INT NOT NULL,
    CONSTRAINT Feedback_PK PRIMARY KEY (FeedbackID),
    CONSTRAINT feedbacktouristFK FOREIGN KEY (UserID) REFERENCES Tourist (UserID),
    CONSTRAINT feedbackresponseIDFK FOREIGN KEY (ResponseID) REFERENCES Response_
(ResponseID)
);

INSERT INTO Feedback_ (FeedbackID, UserID, FeedbackText, FeedbackDate, ResponseID)
VALUES
    ('101', '001', 'Thank you for your feedback!', '2024-05-03', '1'),
    ('102', '002', 'We apologize for the inconvenience.', '2024-04-23', '2'),
    ('103', '003', 'Your suggestions have been noted.', '2024-05-01', '3'),
    ('104', '004', 'We appreciate your input.', '2024-05-02', '4'),
    ('105', '005', 'Please contact us for further assistance.', '2024-04-28', '5');

SELECT * FROM Feedback_;




--------Payment Table --------

CREATE TABLE Payments__ (
    PaymentID INT NOT NULL,
    UserID INT NOT NULL,
    PackageID INT NOT NULL,
    PaymentAmount VARCHAR(255) NOT NULL,
    PaymentMethod VARCHAR(255) NOT NULL,
    PaymentDate DATE NOT NULL,
    ManagerID INT NOT NULL,
    CONSTRAINT Payments__PK PRIMARY KEY (PaymentID),
    CONSTRAINT paymentmanagerFK FOREIGN KEY (ManagerID) REFERENCES Manager (ManagerID),
    CONSTRAINT paymenttouristFK FOREIGN KEY (UserID) REFERENCES Tourist (UserID),
    CONSTRAINT paymentpackageFK FOREIGN KEY (PackageID) REFERENCES Package_ (PackageID)
);

INSERT INTO Payments__ (PaymentID, UserID, PackageID, PaymentAmount, PaymentMethod,
PaymentDate, ManagerID)
VALUES
    ('00001', '001', '01', '46000.00', 'Online', '2024-05-02', '101'),
    ('00002', '002', '02', '40000.00', 'Online', '2024-05-01', '101'),
    ('00003', '003', '03', '42000.00', 'Bank Payment', '2024-04-28', '101'),
    ('00004', '004', '04', '38000.00', 'Online', '2024-05-03', '101'),
    ('00005', '005', '05', '37000.00', 'Bank Payment', '2024-05-03', '101');

SELECT * FROM Payments__;
```

```
--------Report Table --------

CREATE TABLE Report_ (
    ReportID INT NOT NULL,
    ReportDate DATE NOT NULL,
    ManagerID INT NOT NULL,
    AdminID INT NOT NULL,
    CONSTRAINT Report_PK PRIMARY KEY (ReportID),
    CONSTRAINT reportmanagerFK FOREIGN KEY (ManagerID) REFERENCES Manager (ManagerID),
    CONSTRAINT reportadministratorFK FOREIGN KEY (AdminID) REFERENCES Administrator
(AdminID)
);

INSERT INTO Report_ (ReportID, ReportDate, ManagerID, AdminID)
VALUES
    ('1', '2024-05-02', '101', '001'),
    ('2', '2024-05-01', '101', '001'),
    ('3', '2024-04-30', '101', '001'),
    ('4', '2024-04-29', '101', '001'),
    ('5', '2024-04-28', '101', '001');

SELECT * FROM Report_;



--payment and booking table—

CREATE TABLE PaymentBooking (
    PaymentID INT NOT NULL,
    BookingID INT NOT NULL,
    CONSTRAINT PaymentBooking_PK PRIMARY KEY (PaymentID, BookingID),
    CONSTRAINT PaymentBooking_Payment_FK FOREIGN KEY (PaymentID) REFERENCES Payments__
(PaymentID),
    CONSTRAINT PaymentBooking_Booking_FK FOREIGN KEY (BookingID) REFERENCES Booking
(BookingID)
);

INSERT INTO PaymentBooking (PaymentID, BookingID)
VALUES
    ('00001', '0001'),
    ('00002', '0002'),
    ('00003', '0003'),
    ('00004', '0004'),
    ('00005', '0005');

SELECT * FROM PaymentBooking;



--package and booking table—

CREATE TABLE PackageBooking (
    PackageID INT NOT NULL,
    BookingID INT NOT NULL,
    CONSTRAINT PackageBooking_PK PRIMARY KEY (PackageID, BookingID),
    CONSTRAINT PackageBooking_Package_FK FOREIGN KEY (PackageID) REFERENCES Package_
(PackageID),
```

```
    CONSTRAINT PackageBooking_Booking_FK FOREIGN KEY (BookingID) REFERENCES Booking
(BookingID)
);

INSERT INTO PackageBooking (PackageID, BookingID)
VALUES
    ('01', '0001'),
    ('02', '0002'),
    ('03', '0003'),
    ('04', '0004'),
    ('05', '0005');

SELECT * FROM PackageBooking;
```

Performance Requirements

A major role is played by Performance Requirements to make the system successful. They are as follow,

- System must be active 24 hours, 365 days for a registered user to access the system without any inconvenience.
- A Registered user can access the system numerous times by entering his/ her login credentials.
- Login process and loading the pages must be done within few seconds.
- Speed and usability are the performance requirements for this system.
- Registered user can view Hotel details.
- Registered user can edit or delete his/her account details.
- System loads within a minimum time.
- System administrator can manage user accounts.
- Marketing manager and provider can edit/update hotel details.
- System allows manager to approve reservation requests and cancel reservation requests.
- System allows support response to respond to inquiries made by customers.
- System allows support response to add or remove shared experiences and feedback.
- Design user friendly user interface.
- Users must be able to access the website at any time using any device or browser.

Security Requirements

The security section can be pointed out as an important part of every system. A security system is worth it to prevent external influences from entering our system. For a complex system such as Hotel reservation system for tourists, the impact of the security section is strong.

- Personal details of users should be encrypted before send to the database.
- Unauthorized users should be unable to access restricted features.
- The database should have a backup of all the data in the system.
- The password of user account must be a strong password which include uppercase letters, lowercase letters, numbers and special characters.
- For one email address, there should be only one user account.
- Only the administrators can access and modify the data of the system.