MAL2018 INFORMATION MANAGEMENT AND RETRIEVAL REPORT 2

BSSE2409191

Assessment Report 2: TSC HOTEL BOOKING MICROSERVICE

1.Introduction

This report is for Assessment MAL2018 Information and management retrieval coursework 2. The documentation and implementation for the TSC Hotel Booking Microservice. The Microservice allow customer to handle room booking efficiently through user friendly interface while maintaining backend functionality via Restful. Including Database, integration with PHP and secure design.

GitHub Repository	https://github.com/ZULKAMAL12/Assessment-2-Information-management-and-retrieval
Testing	http://risedragon.fwh.is username: user@example.com password: userpassword123

This report details the background of the project, analysis and design considerations, implementation process, evaluation results, and legal, social, ethical and professional issue.

2. Background

This section presents the purpose of the TSC Hotel Booking Micro service and the problem its address. This also to improve scalability, maintainability and security.

TSC Hotel & Tourism need a modern booking system that allow customer to:

- Create account and Login.
- Book a room and mange Booking (edit, booking).
- View room Availability
- Make A payment

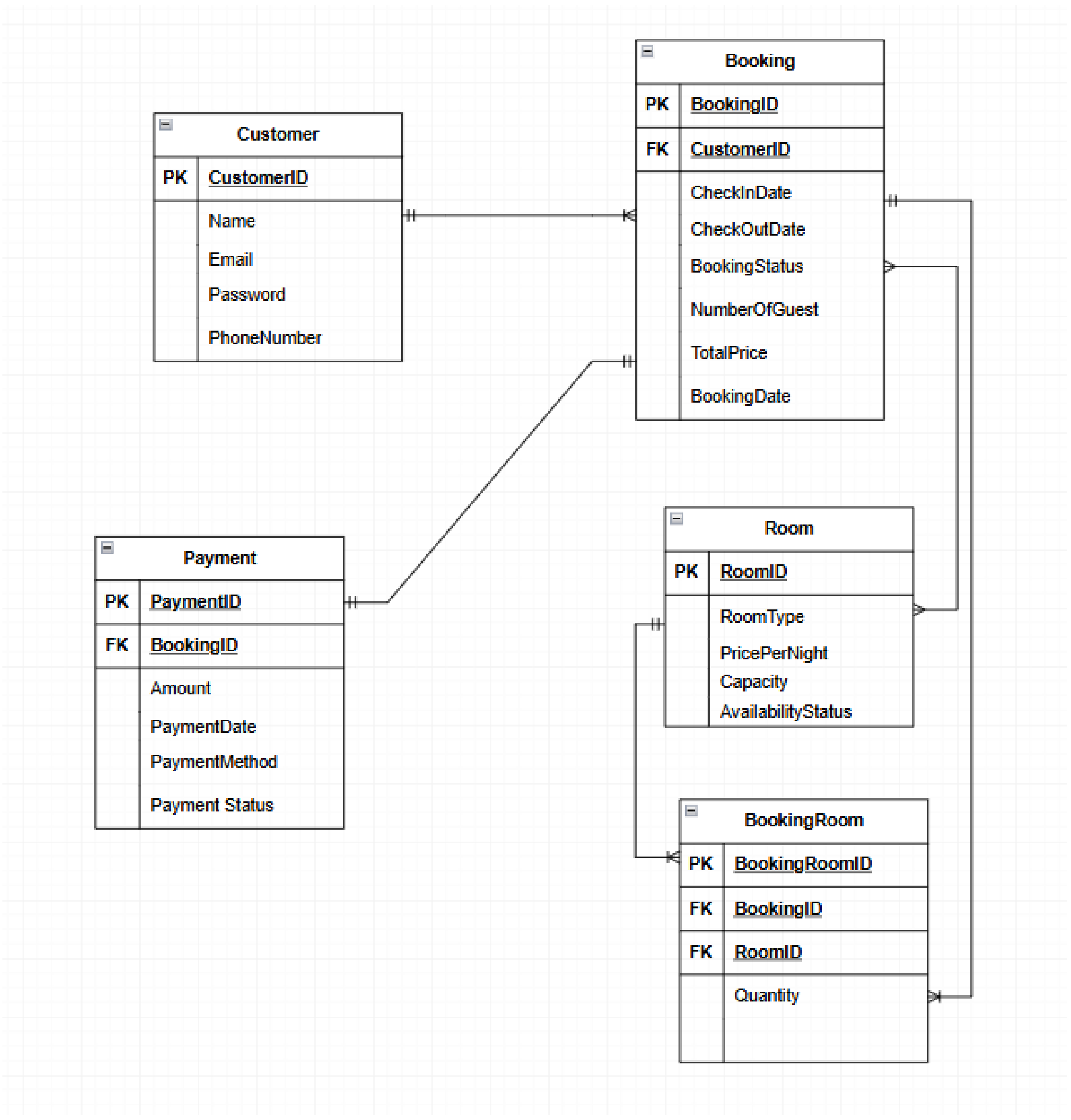
The Database Includes:

- Rooms: Categorized into Family room, Suite and Standard Deluxe.
- Customer: Registered user who can create and manage booking
- Payments: Securely processed and track for all booking

3. Analysis and Design

3.1 Entity-Relationship Diagram (ERD)

Ther ERD below illustrate the relationships between key entities, ensure proper normalization.



Relationships:

• **Customer**-to-**booking**: A customer can make multiple Booking, but each Booking associated with a customer. (One-to-Many).

• **Booking**-to-**Room**: Booking is made for one Room, but one Room can have multiple Booking on different date and times. (Many-to-One).

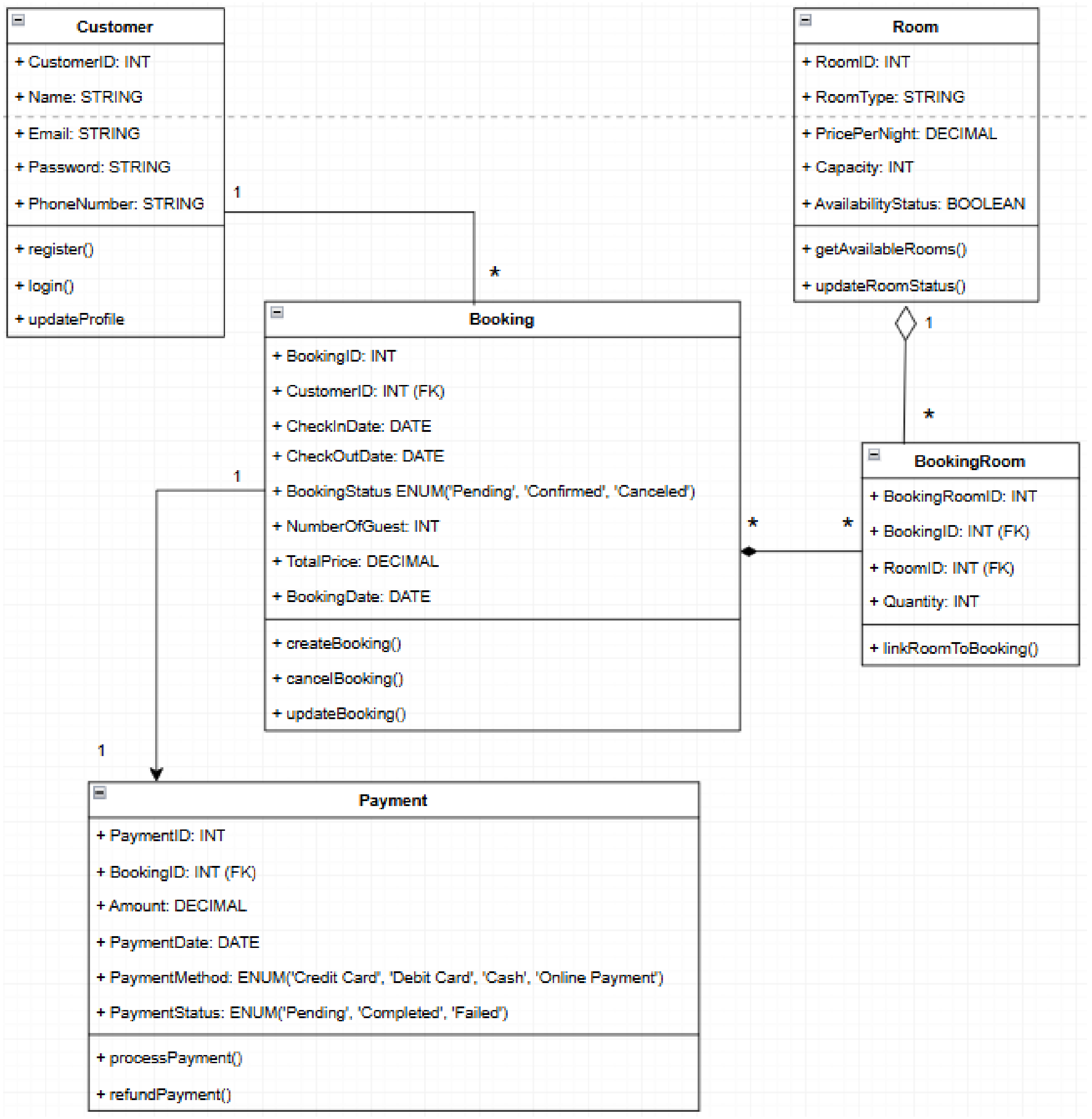
• **Payment**-to-**Booking**: Each Booking can have only one Payment record and each Payment is associated with only one Booking. (One-to-One).

• **Many**-to-**Many** relationship between **Booking** and **Room** resolved via **BookingRoom**.

3.2 UML DIAGRAM

The UML diagram highlight the classes and methods implemented.

- **Customer**: Handle user information, authentication and account management.
- **Room**: Manage room availability.
- **Booking**: Support CRUD operation for Booking.
- **Payment**: Process and track payment
- **BookingRoom**: Link **Rooms** and **Bookings**.



3.3 Database Normalization

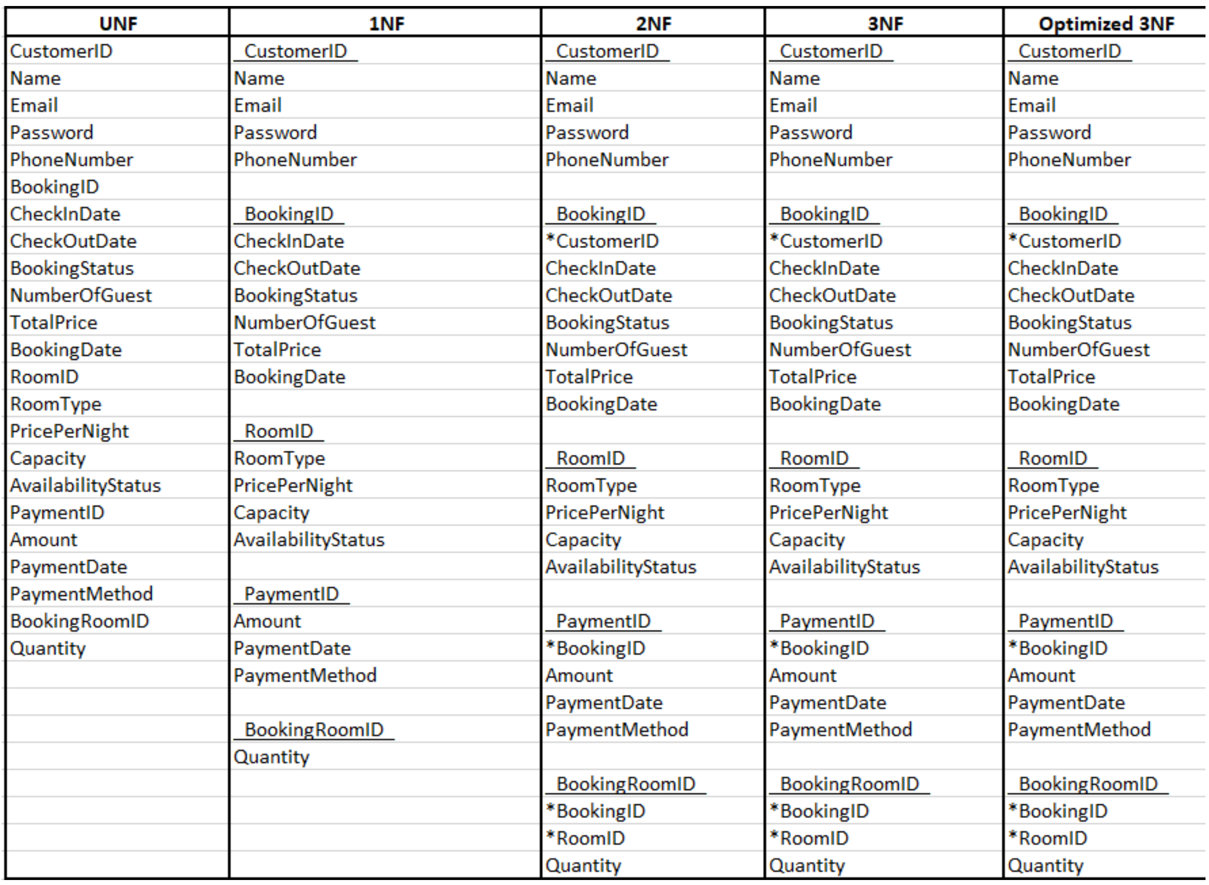
The Database Design

1. UNF: Original dataset with redundant information.

2. 1NF: Remove multi valued attribute.

3. 2NF: Remove partial dependencies.

4. 3NF: Remove transitive dependencies.



UNF	1NF	2NF	3NF	Optimized 3NF
CustomerID	CustomerID	CustomerID	CustomerID	CustomerID
Name	Name	Name	Name	Name
Email	Email	Email	Email	Email
Password	Password	Password	Password	Password
PhoneNumber	PhoneNumber	PhoneNumber	PhoneNumber	PhoneNumber
BookingID				
CheckInDate	BookingID	BookingID	BookingID	BookingID
CheckOutDate	CheckInDate	*CustomerID	*CustomerID	*CustomerID
BookingStatus	CheckOutDate	CheckInDate	CheckInDate	CheckInDate
NumberOfGuest	BookingStatus	CheckOutDate	CheckOutDate	CheckOutDate
TotalPrice	NumberOfGuest	BookingStatus	BookingStatus	BookingStatus
BookingDate	TotalPrice	NumberOfGuest	NumberOfGuest	NumberOfGuest
RoomID	BookingDate	TotalPrice	TotalPrice	TotalPrice
RoomType		BookingDate	BookingDate	BookingDate
PricePerNight	RoomID			
Capacity	RoomType	RoomID	RoomID	RoomID
AvailabilityStatus	PricePerNight	RoomType	RoomType	RoomType
PaymentID	Capacity	PricePerNight	PricePerNight	PricePerNight
Amount	AvailabilityStatus	Capacity	Capacity	Capacity
PaymentDate		AvailabilityStatus	AvailabilityStatus	AvailabilityStatus
PaymentMethod	PaymentID			
BookingRoomID	Amount	PaymentID	PaymentID	PaymentID
Quantity	PaymentDate	*BookingID	*BookingID	*BookingID
	PaymentMethod	Amount	Amount	Amount
		PaymentDate	PaymentDate	PaymentDate
	BookingRoomID	PaymentMethod	PaymentMethod	PaymentMethod
	Quantity			
		BookingRoomID	BookingRoomID	BookingRoomID
		*BookingID	*BookingID	*BookingID
		*RoomID	*RoomID	*RoomID
		Quantity	Quantity	Quantity

4. Implementation

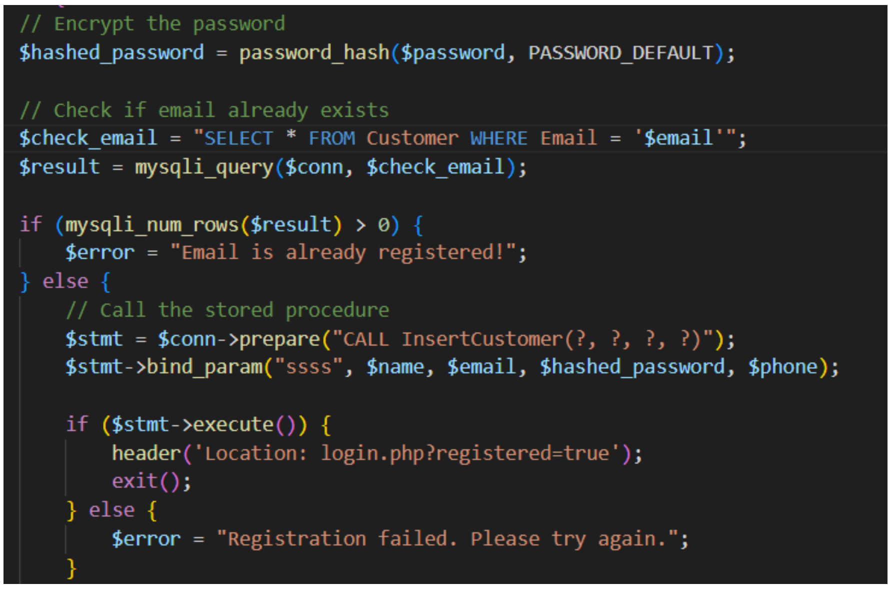
Using Local database xampp

4.1 **Backend**

The backend uses PHP for handling RESTful API request.

- **POST /register**: Register new user.
- **POST /login**: Authentication users.
- **POST /bookings**: Create new booking.
- **PUT /bookings/{id}:** Update an existing booking.
- **DELETE /bookings/{id}:** Cancel a booking

Example Register New User



```php
// Encrypt the password
$hashed_password = password_hash($password, PASSWORD_DEFAULT);

// Check if email already exists
$check_email = "SELECT * FROM Customer WHERE Email = '$email'";
$result = mysqli_query($conn, $check_email);

if (mysqli_num_rows($result) > 0) {
    $error = "Email is already registered!";
} else {
    // Call the stored procedure
    $stmt = $conn->prepare("CALL InsertCustomer(?, ?, ?, ?)");
    $stmt->bind_param("ssss", $name, $email, $hashed_password, $phone);

    if ($stmt->execute()) {
        header('Location: login.php?registered=true');
        exit();
    } else {
        $error = "Registration failed. Please try again.";
    }
```

4.2 Database

The Database Include Five Tables;

1. Customer
2. Room
3. Booking
4. Payment
5. BookingRoom

Customer Table:

- Stores customer information, including hashed passwords.
- Key fields: CustomerID, Name, Email, Password, PhoneNumber.

Room Table:

- Contains details about rooms and their availability.
- Key fields: RoomID, RoomType, PricePerNight, Capacity, AvailabilityStatus.

Booking Table:

- Tracks booking details, linking customers and rooms.
- Key fields: BookingID, CustomerID, CheckInDate, CheckOutDate, TotalPrice.

Payment Table:

- Records payment transactions for bookings.
- Key fields: PaymentID, BookingID, Amount, PaymentDate, PaymentStatus.

BookingRoom Table:

- Resolves the many-to-many relationship between Booking and Room.
- Key fields: BookingRoomID, BookingID, RoomID, Quantity.

Stored procedure handles the key operation like booking creation and payment process. The trigger ensure room availability updated.

4.3 Views and Trigger

- Views: Display active booking with customer room and customer Details.
- Triggers: Automatically update room availability when booking created or cancelled.

5. Evaluation

Testing was conducted using browser-based form submissions.

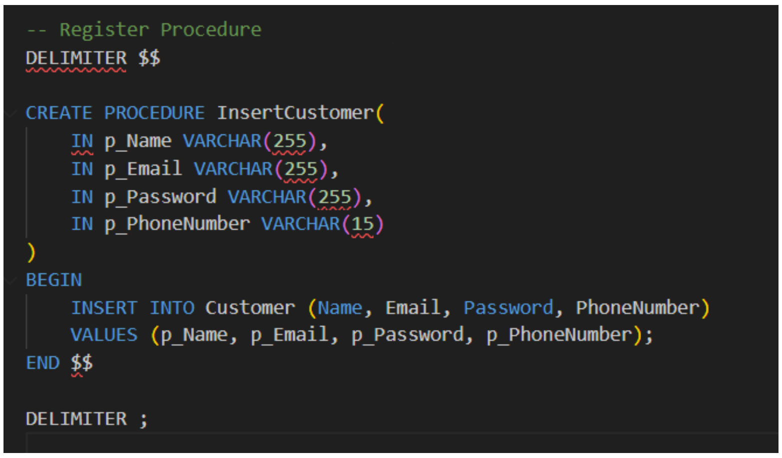
5.1 Testing Result

- User Authentication: Verified secure login/logout process
- CRUD Operations: Ensure functionality for creating, reading, update and deleting bookings.
- Room Availability: Validate automatic updates via trigger

5.2 Observation

- All functional requirement were met.
- Area for improvement
  - Implement token-based authentication
  - Enhance error handling for edge cases.

Procedure Insert Customer Register new User

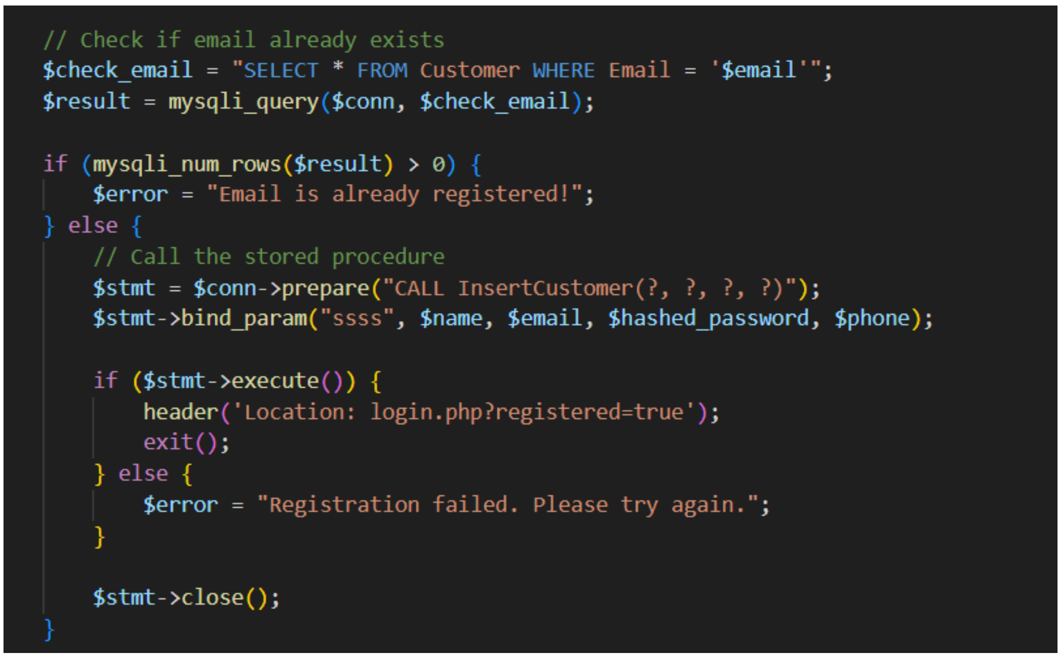


```sql
-- Register Procedure
DELIMITER $$

CREATE PROCEDURE InsertCustomer(
    IN p_Name VARCHAR(255),
    IN p_Email VARCHAR(255),
    IN p_Password VARCHAR(255),
    IN p_PhoneNumber VARCHAR(15)
)
BEGIN
    INSERT INTO Customer (Name, Email, Password, PhoneNumber)
    VALUES (p_Name, p_Email, p_Password, p_PhoneNumber);
END $$

DELIMITER ;
```

Php code to call procedure



```php
// Check if email already exists
$check_email = "SELECT * FROM Customer WHERE Email = '$email'";
$result = mysqli_query($conn, $check_email);

if (mysqli_num_rows($result) > 0) {
    $error = "Email is already registered!";
} else {
    // Call the stored procedure
    $stmt = $conn->prepare("CALL InsertCustomer(?, ?, ?, ?)");
    $stmt->bind_param("ssss", $name, $email, $hashed_password, $phone);

    if ($stmt->execute()) {
        header('Location: login.php?registered=true');
        exit();
    } else {
        $error = "Registration failed. Please try again.";
    }

    $stmt->close();
}
```

6. Legal, Social, Ethical, and Professional

This section addresses the legal, social, ethical, and professional principles observed during the design and implementation of the TSC Hotel Booking Microservice. These considerations ensure compliance with laws, alignment with ethical standards, and professional-grade development practices.

6.1 Legal

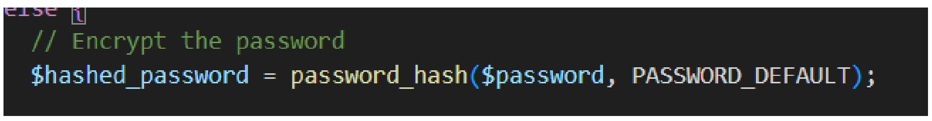
The system compile with international data protection regulation.

- Compliance with Data protection act 2018 ensure secure user data handling.

Implementation:

- Password are hashed using password_hash() to protect user from unauthorized access
- Sensitive customer data is transmitted over HTTPS to prevent interception
- Only minimal personal information is collected to reduce privacy risk.

Example



```
else {
  // Encrypt the password
  $hashed_password = password_hash($password, PASSWORD_DEFAULT);
```

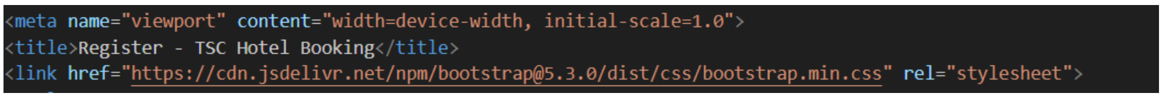
Data Retention Policy

- Booking and payment record are retained only as long as required for legal or business purpose.

6.2 Social Consideration

- Improve user experience through responsive and friendly design and navigation.
- Responsive Design: The use of bootstraps ensure system seamlessly on desktop, tablet and mobile.

Example:



```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Register - TSC Hotel Booking</title>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
```

Transparency:

- All charge, taxes and term clearly display to customer to avoid confusion and misinformation.

Real-time Room Availability

- Customer can check real time room availability, reduce the risk of over booking.
- Implementation
  -The **Room** table includes an **AvailabilityStatus** field that is update dynamically via database trigger.

6.3 Ethical

Ethical principle was followed to ensure fairness, security and respect for user data.

- Secure coding practice prevent misuse of customer data.

Avoiding Bias

- The system ensures fairness by treating users equally with room availability and pricing are determined programmatically without bias.

Environmentally friendly

- By using the system, it reduces the use of paper-based record and support sustainable practice.

6.4 Professional

- Adherence to OWASP top 10 guideline.

Future proofing

- The use of modular code ensure that the system can be easily updated and maintained.

Example of security and professional implementation

1.Password handling

- Password hashed using php password_hash() and verify using password_verify() when login.
  Example:

```php
    $hashed_password = password_hash($password, PASSWORD_BCRYPT);
if (password_verify($password, $hashed_password)) {
    echo "Login successful!";
} else {
    echo "Invalid credentials.";
}
```

Privacy

- Password using password_hash () before storage.
- Sensitive user and payment data is encrypted during transmission using HTTPS.

Conclusion for LSEP

The consideration ensure that the TSC Booking Microservice is legally compliant, responsible socially, ethical and professional. The project adheres to secure development standard while prioritizing user satisfaction.

7. Conclusion

The report Outline the successful implementation of the TSC Hotel Booking Microservice. The project demonstration a database design, secure backend functionality and user-friendly feature. Future enhancement including adding advance security and integration notification.

Reference:

1. W3Schools (2025) ‘PHP MySQL Integration’, *W3Schools*. Available at: https://www.w3schools.com/php/php_mysql_intro.asp (Accessed: 1 January 2024).
2. YouTube *PHP Tutorial for Beginners - Step by Step Guide*. Available at: https://www.youtube.com/watch?v=MV8AT9a2oSM&list=PLWxTHN2c_6cbh1C7yIskoXszoTl-okogt (Accessed: 20 December 2025).
3. InfinityFree (2025) *Free Hosting Platform*. Available at: https://infinityfree.net/ (Accessed: 1 January 2025).
4. OWASP Foundation (2024) *OWASP Top Ten Security Risks*. Available at: https://owasp.org/www-project-top-ten/ (Accessed: 23 December 2025).