BANGLADESH UNIVERSITY OF PROFESSIONALS (BUP)

Assignment on Queries

Tentative Title: **Beauty Salon Management**

1.

Student Name	Student ID
Shamoyeta Mourin Mouly	2252421036
Latifa Nishat Nishi	2252421062
Tahsina Tabassum Roza	2252421084
Raiyan Bin Sarwar	2252421096

2.	Name of the Department/Office/Faculty:		Faculty of Science and Technology (FST)
3.	Name of the University:		Bangladesh University of Professional
4.	Title of the Project:		Beauty Salon Management
6.	Place where the work will be performed:		
	● Name of the University		Bangladesh University of Professional
	● Name of the Department/Institute		Department of Computer Science and Engineering (CSE)

Beauty Salon Management

Introduction

The Beauty Salon Management System streamlines daily salon operations through an efficient database management system. This project implements a comprehensive set of SQL queries to handle essential functions like user authentication, appointment scheduling, inventory management, and reporting.

The system is built with a multi-tiered access approach, serving different user roles including administrators, receptionists, staff members, and customers. Each role has specific functionalities and access levels, ensuring secure and efficient management of salon operations. The implementation uses MySQL queries optimized for performance and reliability, with prepared statements to prevent SQL injection and ensure data security

Queries:

The following documentation details the core database queries that power each system module:

LogIn

These queries handle user authentication for different roles in the system.

i. Admin Login Check

```
$sql_admin = "SELECT * FROM admin WHERE admin_email='$login_input'";

$sql_admin = "SELECT * FROM admin WHERE admin_phone='$login_input'";
```

These queries check if the provided email or phone number matches an entry in the admin table to verify if the user is an admin.

ii. Customer Login Check

```
$sql_customer = "SELECT * FROM customer WHERE cust_mail='$login_input'";

$sql_customer = "SELECT * FROM customer WHERE cust_phone='$login_input'";
```

These queries check the customer table for a match with the provided email or phone number to verify if the user is a customer.

iii. Staff Login Check

```
$sql_staff = "SELECT * FROM staff WHERE staff_mail='$login_input'";

$sql_staff = "SELECT * FROM staff WHERE staff_phone='$login_input'";
```

These queries check the staff table for a match with the provided email or phone number to verify if the user is a staff member.

iv. **Receptionist Login Check**

If the staff role is 'Receptionist', the user is redirected to the receptionist dashboard:

```
if ($staff['staff_role'] == 'Receptionist') {
    header("Location: receptionist_dashboard.php");
}
```

Customer Signup

i. **Generate Customer ID**

```
$cust_query = "SELECT MAX(CAST(SUBSTRING(cust_id, 2) AS UNSIGNED)) as max_id
FROM customer";
```

This query generates a new customer ID by finding the maximum existing customer ID and incrementing it.

ii. **Insert Customer**

```
$stmt = $conn->prepare("INSERT INTO customer (cust_id, cust_name,
cust_phone, cust_mail, cust_loc) VALUES (?, ?, ?, ?, ?)");
```

This query inserts a new customer record into the customer table with the provided details.

Admin + *Receptionist* Profile Management

1. Admin Management

i. **Search Admins**

```
$query = "SELECT * FROM admin WHERE $category LIKE '%$search%'";

$query = "SELECT * FROM admin";
```

These queries search for admins based on the selected category and search term, or fetch all admin records if no search term is provided.

ii. **Generate Admin ID**

```
$admin_id_query = "SELECT MAX(CAST(SUBSTRING(admin_id, 2) AS UNSIGNED)) AS
max_id FROM admin";
```

This query generates a new admin ID by finding the maximum existing admin ID and incrementing it.

iii. **Insert New Admin**

```
$insert_query = "INSERT INTO admin (admin_id, admin_name, admin_email, admin_phone) VALUES ('$admin_id', '$admin_name', '$admin_email', '$admin_phone')";
```

This query inserts a new admin record into the database.

iv. **Delete Admin**

```
$delete_query = "DELETE FROM admin WHERE admin_id = '$delete_id'";
```

This query deletes an admin record based on the provided admin ID.

2. Customer Management

i. **Search Customers**

```
$query = "SELECT * FROM customer WHERE $category LIKE '%$search%'";
```

This query searches for customers based on the selected category (e.g., cust_id, cust_name, cust_phone, cust_mail, cust_loc) and the search term provided by the user.

ii. **Default Query**

```
$query = "SELECT * FROM customer";
```

This is the default query used when no search term or category is provided, fetching all customer records.

iii. **Generate Customer ID**

```
$cust_query = "SELECT MAX(CAST(SUBSTRING(cust_id, 2) AS UNSIGNED)) AS max_id FROM customer";
```

This query generates a new customer ID by finding the maximum existing customer ID and incrementing it.

iv. **Insert Customer**

```
$insert_query = "INSERT INTO customer (cust_id, cust_name, cust_phone, cust_mail, cust_loc) VALUES ('$cust_id', '$cust_name', '$cust_phone', '$cust_mail', '$cust_loc')";
```

This query inserts a new customer record into the customer table with the provided details.

v. **Delete Customer**

```
$delete_query = "DELETE FROM customer WHERE cust_id = '$delete_id'";
```

This query deletes a customer record based on the provided customer ID.

3. Staff Management

i. **Search Staff**

```
$query = "SELECT * FROM staff WHERE $category LIKE '%$search%'";

$query = "SELECT * FROM staff";
```

These queries search for staff members based on the selected category and search term, or fetch all staff records if no search term is provided.

ii. **Generate Staff ID**

```
$prod_query = "SELECT MAX(CAST(SUBSTRING(staff_id, 2) AS UNSIGNED)) as max_id FROM staff";
```

This query generates a new staff ID by finding the maximum existing staff ID and incrementing it.

iii. **Insert New Staff**

```
$stmt = $conn->prepare("INSERT INTO staff (staff_id, staff_name, staff_phone, staff_mail, staff_role) VALUES (?, ?, ?, ?, ?)");
```

This query inserts a new staff record into the database.

iv. **Delete Staff**

```
$delete_query = "DELETE FROM staff WHERE staff_id = '$delete_id'";
```

This query deletes a staff record based on the provided staff ID.

4. Appointment Details Management

i. **Search Appointments**

```
$query = "

    SELECT DISTINCT

        a.app_id,

        c.cust_name,

        c.cust_phone,

        s.sname as service_name,
        s.sduration as service_duration,

        st.staff_name,

        st.staff_role,

        ad.status,

        ad.app_date,

        TIME_FORMAT(s.sduration, '%H:%i') as formatted_duration

    FROM appointment a

    JOIN customer c ON a.cust_id = c.cust_id

    JOIN appointment_services aps ON a.app_id = aps.app_id

    JOIN services s ON aps.sid = s.sid

    JOIN staff st ON aps.staff_id = st.staff_id

    JOIN appointment_details ad ON a.app_id = ad.app_id

    WHERE 1=1";
```

This query retrieves appointment details, including customer information, service details, staff information, and appointment status.

ii. **Filter by Date**

```
$query .= " AND DATE(ad.app_date) = ?";
```

This query filters appointments by the specified date.

iii. **Filter by Service Category**

```
$query .= " AND s.sname = ?";
```

This query filters appointments by the specified service category.

iv. **Search by Term**

```
$query .= " AND (c.cust_name LIKE ? OR c.cust_phone LIKE ? OR st.staff_name
LIKE ?)";
```

This query searches appointments by customer name, customer phone, or staff name.

v. **Sort Appointments**

```
$query .= " ORDER BY a.app_id " . ($sort_order === 'ASC' ? 'ASC' : 'DESC');
```

This query sorts appointments by appointment ID in ascending or descending order.

vi. **Get All Services for Dropdown**

```
$services_query = "SELECT DISTINCT sname FROM services ORDER BY sname";
```

This query retrieves all distinct service names for the dropdown menu.

vii. **Prepare and Execute Main Query**

```
$stmt = $conn->prepare($query);

if (!empty($params)) {

    $stmt->bind_param($param_types, ...$params);

}

$stmt->execute();

$result = $stmt->get_result();
```

This code prepares and executes the main query with the specified parameters.

5. Appointment Confirmation Management

i. **Search Appointments**

```
$query = "SELECT * FROM appointment_details WHERE $category LIKE '%$search%'
ORDER BY app_date";

$query = "SELECT * FROM appointment_details";
```

These queries search for appointment details based on the selected category and search term, or fetch all appointment records sorted by date if no search term is provided.

ii. **Update Appointment Status**

```
$update_query = "UPDATE appointment_details SET status = '$status' WHERE
pay_id = '$pay_id'";
```

This query updates the status of an appointment based on the provided payment ID.

iii. **Delete Appointment**

```
$delete_query = "DELETE FROM appointment WHERE app_id = '$delete_id'";
```

This query deletes an appointment record based on the provided appointment ID.

6. Staff Schedules

i. **Search Staff Schedules**

```
$query = "
    SELECT
        s.staff_id,
        s.staff_name,
        s.staff_role,
        a.app_id,
        a.app_date,
        c.cust_name,
        sv.sname,
        sv.sduration
    FROM staff s
    LEFT JOIN appointment_services as_srv ON s.staff_id = as_srv.staff_id
    LEFT JOIN appointment a ON as_srv.app_id = a.app_id
    LEFT JOIN services sv ON as_srv.sid = sv.sid
    LEFT JOIN customer c ON a.cust_id = c.cust_id
    LEFT JOIN appointment_details ad ON a.app_id = ad.app_id
    WHERE a.app_date = ? AND ad.status = 'Confirmed'";
```

This query retrieves staff schedules, including staff information, appointment details, customer information, and service details for confirmed appointments on the selected date.

ii. **Filter by Search Query**

```
$query .= " AND (s.staff_id LIKE ? OR s.staff_name LIKE ? OR s.staff_role
LIKE ?)";
```

This query filters staff schedules based on the search query, which can be staff ID, staff name, or staff role.

iii. **Sort Staff Schedules**

```
$query .= " ORDER BY s.staff_name, sv.sname";
```

This query sorts staff schedules by staff name and service name.

iv. **Prepare and Execute Main Query**

```
$stmt = $conn->prepare($query);


if (!empty($searchQuery)) {
    $searchParam = "%$searchQuery%";
  $stmt->bind_param("ssss", $selectedDate, $searchParam, $searchParam, $searchParam);
} else {
  $stmt->bind_param("s", $selectedDate);
}


$stmt->execute();
$result = $stmt->get_result();
```

This code prepares and executes the main query with the specified parameters.

7. Services Management

i. **Search Services**

```
$query = "SELECT * FROM services WHERE $category LIKE '%$search%'";
$query = "SELECT * FROM services";
```

These queries search for services based on the selected category and search term, or fetch all service records if no search term is provided.

ii. **Generate Service ID**

```
$sid_query = "SELECT MAX(CAST(SUBSTRING(sid, 2) AS UNSIGNED)) AS max_id FROM services";
```

This query generates a new service ID by finding the maximum existing service ID and incrementing it.

iii. **Insert New Service**

```
$insert_query = "INSERT INTO services (sid, sname, sprice, sduration) VALUES ('$sid', '$sname', '$sprice', '$formatted_duration')";
```

This query inserts a new service record into the database.

iv. **Delete Service**

```
$delete_query = "DELETE FROM services WHERE sid = '$delete_id'";
```

This query deletes a service record based on the provided service ID.

8. Sales Report

i. **Generate Sales Report**

```
$query = "SELECT
            s.sname,
            s.sprice as service_price,
            COUNT(DISTINCT asvc.app_id) as service_count,
            SUM(s.sprice) as total_sales
          FROM services s
          INNER JOIN appointment_services asvc ON s.sid = asvc.sid
          INNER JOIN appointment_details ad ON asvc.app_id = ad.app_id
          WHERE ad.status = 'Confirmed'";
```

This query retrieves sales data, including service name, service price, number of sales, and total sales amount for confirmed appointments.

ii. **Filter by Date**

```
$query .= " AND DATE(ad.app_date) = '$selected_date'";
```

This query filters sales data by the selected date.

iii. **Filter by Month**

```
$query .= " AND MONTH(ad.app_date) = '$month_num' AND YEAR(ad.app_date) =
'$year'";
```

This query filters sales data by the selected month and year.

iv. **Filter by Year**

```
$query .= " AND YEAR(ad.app_date) = '$selected_year'";
```

This query filters sales data by the selected year.

v. **Group and Order Sales Data**

```
$query .= " GROUP BY s.sid, s.sname, s.sprice
            ORDER BY s.sname";
```

This query groups sales data by service ID, service name, and service price, and orders the results by service name.

9. Staff Popularity Report

i. **Generate Staff Popularity Report**

```
$query = "SELECT
            s.staff_id,
            s.staff_name,
            s.staff_role,
            COALESCE(COUNT(DISTINCT CASE
                WHEN ad.status = 'Confirmed' THEN as2.app_id
                END), 0) as demand
          FROM staff s
          LEFT JOIN appointment_services as2 ON s.staff_id = as2.staff_id
          LEFT JOIN appointment_details ad ON as2.app_id = ad.app_id";
```

This query retrieves staff popularity data, including staff ID, staff name, staff role, and the number of confirmed appointments.

ii. **Filter by Category**

```
$whereClauses[] = "s.staff_role = '$category'";
```

This query filters staff popularity data by the selected staff role category.

iii. **Filter by Date**

```
$whereClauses[] = "DATE(ad.app_date) = '$selected_date'";
```

This query filters staff popularity data by the selected date.

iv. **Filter by Month**

```
$whereClauses[] = "MONTH(ad.app_date) = '$month_num' AND YEAR(ad.app_date)
= '$year'";
```

This query filters staff popularity data by the selected month and year.

v. **Filter by Year**

```
$whereClauses[] = "YEAR(ad.app_date) = '$selected_year'";
```

This query filters staff popularity data by the selected year.

vi. **Group and Order Staff Popularity Data**

```
$query .= " GROUP BY s.staff_id, s.staff_name, s.staff_role
            ORDER BY demand DESC, s.staff_name ASC";
```

This query groups staff popularity data by staff ID, staff name, and staff role, and orders the results by demand and staff name.

10.Inventory Management System

i. **Get Inventory Status**

```
$query = "SELECT
            i.product_id,
            i.product_name,
            i.product_category,
            i.quantity as current_stock,
            i.price,
            SUM(COALESCE(is2.quantity_used, 0)) as total_used,
            i.quantity - SUM(COALESCE(is2.quantity_used, 0)) as
remaining_stock
          FROM inventory i
          LEFT JOIN inventory_services is2 ON i.product_id = is2.product_id
          GROUP BY i.product_id";
```

This query retrieves the current inventory status, including product ID, product name, product category, current stock, price, total used quantity, and remaining stock.

ii. **Get Usage History**

```
$query = "SELECT
            is2.appointment_date,
            i.product_name,
            i.product_category,
            s.sname as service_name,
            is2.quantity_used,
            c.cust_name,
            st.staff_name
          FROM inventory_services is2
          JOIN inventory i ON is2.product_id = i.product_id
          JOIN services s ON is2.sid = s.sid
          JOIN appointment a ON is2.app_id = a.app_id
          JOIN customer c ON a.cust_id = c.cust_id
          JOIN appointment_services aps ON is2.app_id = aps.app_id AND
is2.sid = aps.sid
          JOIN staff st ON aps.staff_id = st.staff_id
          ORDER BY is2.appointment_date DESC";
```

This query retrieves the usage history, including appointment date, product name, product category, service name, quantity used, customer name, and staff name.

iii. **Get Product Names**

```
$query = "SELECT DISTINCT product_name FROM inventory ORDER BY
product_name";
```

This query retrieves all distinct product names from the inventory.

iv. **Update Inventory**

```
$query = "UPDATE inventory SET quantity = :quantity WHERE product_id =
:product_id";
```

This query updates the quantity of a specific product in the inventory based on the provided product ID.

11.Inventory Updation

i. **Search Inventory**

```
$query = "SELECT * FROM inventory WHERE $category LIKE '%$search%'";

$query = "SELECT * FROM inventory";
```

These queries search for inventory items based on the selected category and search term, or fetch all inventory records if no search term is provided.

ii. **Generate Product ID**

```
$prod_query = "SELECT MAX(CAST(SUBSTRING(product_id, 2) AS UNSIGNED)) as max_id FROM inventory";
```

This query generates a new product ID by finding the maximum existing product ID and incrementing it.

iii. **Insert New Inventory Item**

```
$stmt = $conn->prepare("INSERT INTO inventory (product_id, product_name, product_category, quantity, price, supplier_name, supplier_contact) VALUES (?, ?, ?, ?, ?, ?, ?)");
```

This query inserts a new inventory item into the database.

iv. **Delete Inventory Item**

```
$delete_query = "DELETE FROM inventory WHERE product_id = ?";
```

This query deletes an inventory item based on the provided product ID.

12.Review Management

i. **Search Reviews**

```
$query = "SELECT review.rev_id, customer.cust_name, review.rating, review.comment

          FROM review

          JOIN customer ON review.cust_id = customer.cust_id

          WHERE $where_clause";
```

This query retrieves reviews based on the search term and selected category, joining the review and customer tables to get customer names.

ii. **Delete Review**

```
$delete_query = "DELETE FROM review WHERE rev_id = '$delete_id'";
```

This query deletes a review record based on the provided review ID.

Staff Profile Management

1. Staff Dashboard

i. **Retrieve Staff Details**

```
$sql = "SELECT * FROM staff WHERE staff_id='$staff_id'";
```

This query retrieves the details of the logged-in staff member based on their staff ID.

ii. **Retrieve Upcoming Appointments**

```
$appointments_query = "SELECT appointment.app_id, appointment.app_date, customer.cust_id, customer.cust_name, services.sname

                       FROM appointment

                       JOIN appointment_services ON appointment.app_id = appointment_services.app_id

                       JOIN services ON appointment_services.sid = services.sid

                       JOIN customer ON appointment.cust_id = customer.cust_id

                       WHERE appointment_services.staff_id = '$staff_id'

                       AND appointment.app_date >= CURDATE()

                       ORDER BY appointment.app_date ASC";
```

This query retrieves the upcoming appointments for the logged-in staff member, including appointment ID, appointment date, customer ID, customer name, and service name.

Customer Profile Management

1. Services Page

i. **Fetch Services**

```
$sql = "SELECT * FROM services";
```

This query retrieves all services from the services table in the database.

ii. **Display Services**

```
while ($row = $result->fetch_assoc()) {
    echo '<div class="service-card" data-category="' . $row["category"] . '">';
    echo '<h3>' . $row["sname"] . '</h3>';
    echo '<p>Price: ' . $row["sprice"] . ' BDT</p>';
    echo '<p>Duration: ' . $row["sduration"] . '</p>';
    echo '<a href="book_appointment.php" class="btn">Book Appointment</a>';
    echo '</div>';
}
```

This code displays each service in a card format, showing the service name, price, duration, and a link to book an appointment.

iii. **Filter Services**

```
function filterServices(category) {
    const searchInput = document.getElementById('search').value.toLowerCase();
    const cards = document.querySelectorAll('.service-card');
    cards.forEach(card => {
        const serviceName = card.querySelector('h3').innerText.toLowerCase();
        if ((category === 'all' || card.getAttribute('data-category') === category) && serviceName.includes(searchInput)) {
            card.style.display = 'block';
        } else {
            card.style.display = 'none';
        }
    });
}
```

This JavaScript function filters the displayed services based on the selected category and search input.

2. Book an Appointment

i. **Fetch Services**

```
$sql = "SELECT * FROM services";

$result = $conn->query($sql);
```

This query retrieves all services from the services table in the database.

ii. **Fetch Staff**

```
$sql = "SELECT * FROM staff";

$result = $conn->query($sql);
```

This query retrieves all staff members from the staff table in the database.

iii. **Check if Customer Exists**

```
$sql = "SELECT cust_id FROM customer WHERE cust_phone='$phone' AND cust_mail='$email'";

$result = $conn->query($sql);
```

This query checks if a customer with the provided phone number and email already exists in the customer table.

iv. **Insert New Customer**

```
$sql = "INSERT INTO customer (cust_id, cust_name, cust_phone, cust_mail) VALUES ('$cust_id', '$name', '$phone', '$email')";

$conn->query($sql);
```

This query inserts a new customer record into the customer table.

v. **Check Staff Availability**

```
$sql = "SELECT COUNT(*) as count FROM appointment_staff WHERE staff_id='$staff' AND app_id IN (SELECT app_id FROM appointment WHERE app_date='$app_date')";

$result = $conn->query($sql);
```

This query checks if the selected staff member has less than 3 appointments on the selected date.

vi. **Assign Free Staff Member**

```
$sql = "SELECT staff_id FROM staff WHERE staff_id NOT IN (SELECT staff_id FROM appointment_staff WHERE app_id IN (SELECT app_id FROM appointment WHERE app_date='$app_date')) LIMIT 1";

$result = $conn->query($sql);
```

This query assigns a free staff member if no specific staff member is selected.

vii. **Insert Appointment**

```
$sql = "INSERT INTO appointment (app_id, app_date, cust_id) VALUES ('$app_id', '$app_date', '$cust_id')";

$conn->query($sql);
```

This query inserts a new appointment record into the appointment table.

viii. **Insert Appointment Details**

```
$sql = "INSERT INTO appointment_details (pay_id, app_id, cust_id, pay_date, pay_method, total_bill, status, app_date) VALUES ('$pay_id', '$app_id', '$cust_id', NOW(), '$pay_method', '$total_bill', 'Pending', '$app_date')";

$conn->query($sql);
```

This query inserts appointment details into the appointment_details table.

ix. **Insert Appointment Services**

```
foreach ($services as $service) {

    $sql = "INSERT INTO appointment_services (app_id, sid, staff_id) VALUES ('$app_id', '$service', '$staff')";

  $conn->query($sql);

}
```

This query inserts the selected services for the appointment into the appointment_services table.

x. **Insert Appointment Staff**

```
$sql = "INSERT INTO appointment_staff (app_id, staff_id) VALUES ('$app_id', '$staff')";

$conn->query($sql);
```

This query inserts the assigned staff member for the appointment into the appointment_staff table.

3. Appointment History

i. **Fetch Customer ID**

```
$sql = "SELECT cust_id FROM customer WHERE cust_phone='$phone'";

$result = $conn->query($sql);
```

This query retrieves the customer ID based on the provided phone number.

ii. **Fetch Appointment History**

```
$sql = "SELECT a.app_id, a.app_date, ad.total_bill, ad.status, s.sname, s.sprice, st.staff_name

        FROM appointment a

        JOIN appointment_details ad ON a.app_id = ad.app_id

        JOIN appointment_services aps ON a.app_id = aps.app_id

        JOIN services s ON aps.sid = s.sid

        JOIN staff st ON aps.staff_id = st.staff_id

        WHERE a.cust_id = '$cust_id'";

$result = $conn->query($sql);
```

This query retrieves the appointment history for the customer, including appointment ID, appointment date, total bill, status, service name, service price, and staff name.

iii. **Cancel Appointment**

```
$sql = "DELETE FROM appointment WHERE app_id='$app_id'";

if ($conn->query($sql) === TRUE) {

    echo "<p>Appointment cancelled successfully!</p>";

} else {

  echo "<p>Error cancelling appointment: " . $conn->error . "</p>";
}
```

This query deletes an appointment based on the provided appointment ID.

4. Give a Review

i. **Fetch Customer Details**

```
$sql = "SELECT cust_name, cust_id FROM customer WHERE cust_id='$user_id'";

$result = $conn->query($sql);
```

This query retrieves the customer details based on the logged-in user ID.

ii. **Submit Review**

```
$sql = "INSERT INTO review (rev_id, cust_id, rating, comment) VALUES ('$rev_id', '$cust_id', '$rating', '$comment')";

if ($conn->query($sql) === TRUE) {

    echo "<p class='success-message'>Review submitted successfully!</p>";

} else {

  echo "<p class='error-message'>Error: " . $sql . "<br>" . $conn->error . "</p>";

}
```

This query inserts a new review into the review table with the provided rating and comment.

Conclusion

The implemented query structure effectively supports all core salon management functions while maintaining data integrity and security. The system successfully handles user authentication, appointment management, inventory tracking, and business reporting, creating a robust foundation for salon operations.

Future enhancements could include query optimization for larger datasets and additional reporting capabilities while maintaining the current database structure.