Mini project report on Amusement park DBMS

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering

UE23CS351A – DBMS Project

DBMS Mini Project Report

Name: N.S Likhith Chandra SRN: PES2UG23CS366

Name: Nikitha Devaraj SRN: PES2UG23CS388

under the guidance of

Prof. Shilpa S

Assistant Professor

PES University

AUG - DEC 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

CERTIFICATE

This is to certify that the mini project entitled

Amusement Park Management System

is a bonafide work carried out by

Name: N.S Likhith Chandra ***SRN: PES2UG23CS366***

Name: Nikitha Devaraj ***SRN:PES2UG23CS388***

In partial fulfilment for the completion of fifth semester DBMS Project (UE20CSS301) in the Program of Study -Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period AUG. 2025 – DEC. 2025. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The project has been approved as it satisfies the 5th semester academic requirements in respect of project work.

Signature

Prof. Shilpa S

Assistant Professor

DECLARATION

We hereby declare that the DBMS Project entitled **Amusement Park Management System** has been carried out by us under the guidance of

Prof. Shilpa S , Assistant Professor and submitted in partial fulfilment of the course requirements for the award of degree of

Bachelor of Technology

in

Computer Science and Engineering

of

PES University, Bengaluru

during the academic semester AUG – DEC 2025.

N S Likhith Chandra **PES2UG23CS366**

[illegible]

Nikitha Devaraj **PES2UG23CS388**

Nikitha

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Shilpa S, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE21CS351 - DBMS Project.

I take this opportunity to thank Dr. Sandesh B J, C, Professor,ChairPerson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this DBMS Project could not have been completed without the continual support and encouragement I have received from my family and friends.

PROJECT TITLE:

Funderland-Where the Fun Begins!

Amusement Park Management System

PROBLEM STATEMENT:

Design and implement a comprehensive web-based management system for an amusement park, enabling users and visitors to seamlessly register, authenticate, book tickets, purchase cards, provide feedback, and interact with mess/food and rides/shop modules. The system must support distinct user and visitor roles, offer administrative controls for managing items and user bookings, and ensure secure, reliable storage and retrieval of data across all operations using a relational database.

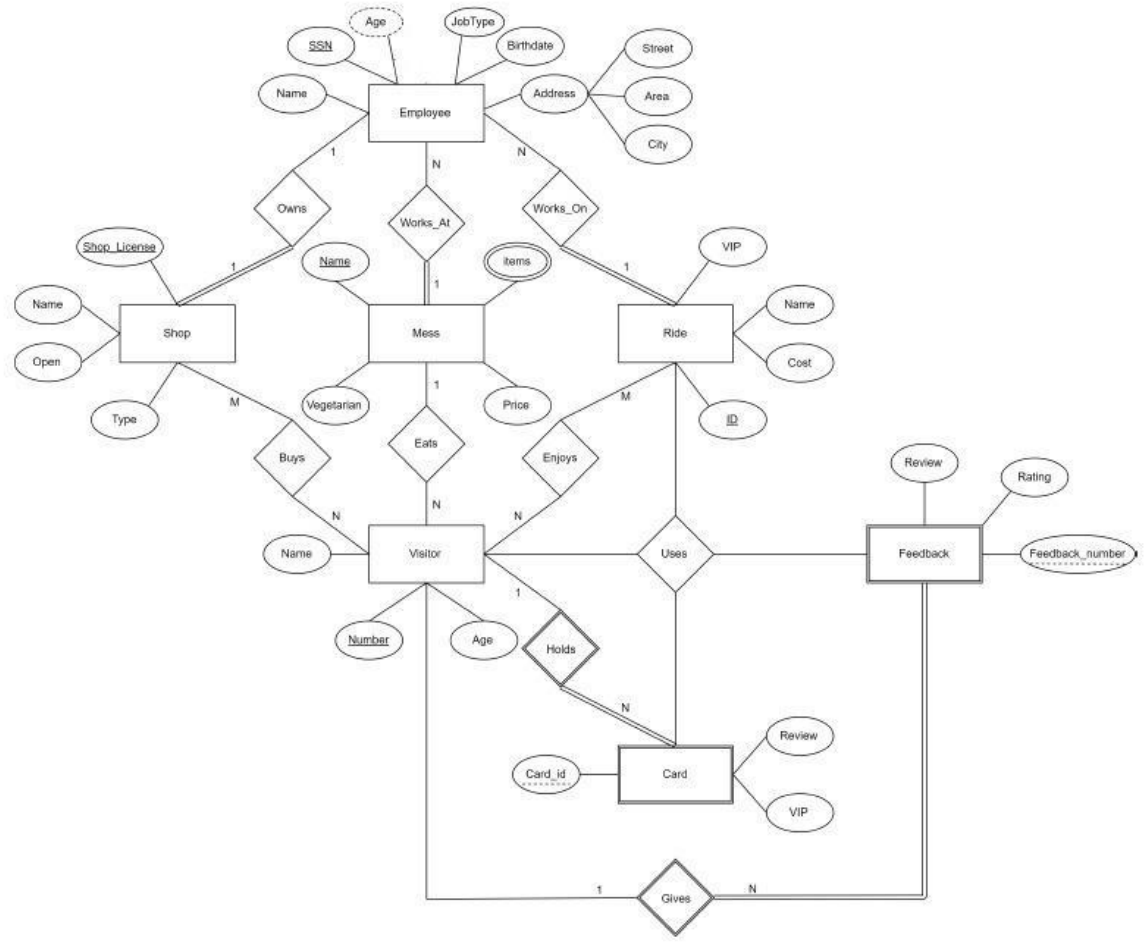
USER REQUIREMENT SPECIFICATION:

- Users must be able to create an account, log in securely, and manage their profile information.
- The system should allow users to book amusement park tickets, purchase park cards, and view their past transactions.
- Users can submit feedback regarding their park experience directly through the website interface.
- Both mess/food and rides/shop sections should allow users to browse available items and see details (prices, availability).
- Separate views should be provided for registered users (with booking/purchase privileges) and visitors (limited browsing access).
- Administrative users or staff should be able to manage mess/food items, rides, shop items, bookings, purchases, and view submitted feedback.
- Data integrity, privacy, and consistent validation of inputs (such as email and password strength) must be enforced across all user-facing forms.
- The system must ensure that CRUD operations (Create, Read, Update, Delete) are fully supported for relevant entities through both frontend UI and backend logic.
- Immediate updates and clear feedback messages should be provided for user actions (e.g., successful booking, purchase confirmation, feedback submission, and error handling).
- Every transaction and feedback entry should be stored in the database with timestamps for auditing and reporting purposes.
- Users must be able to log out at any time, and session management should prevent unauthorized access.

SOFTWARES/TOOLS/PROGRAMMING LANGUAGES USED:

- XAMPP: Provides the Apache server, MySQL database, and PHP environment for development and local deployment.
- MySQL Workbench: Used for database design, schema creation, and SQL commands.
- PHP: Backend scripting language for server-side processing, session management, and database interaction.
- HTML & CSS: For building and styling the website frontend, forms, and user interfaces.
- JavaScript: Adds interactivity, validates user inputs, and enhances responsive interface behavior.
- Web Browser (Chrome, Firefox, etc.): For testing and presentation of the web application.
- Additional tools (if used): VS Code for asset creation.

ER DIAGRAM:



Age
JobType
SSN
Birthdate
Street
Name
Employee
Address
Area
City
1
N
N
Owns
Works_At
Works_On
Shop_License
Name
items
VIP
1
1
1
Name
Shop
Mess
Ride
Name
Open
Cost
M
1
M
Type
Vegetarian
Price
ID
Buys
Eats
Enjoys
Review
Rating
N
N
N
Name
Visitor
Uses
Feedback
Feedback_number
1
Number
Age
Holds
N
Review
Card_id
Card
VIP
1
Gives
N

Relational Schema Diagram:

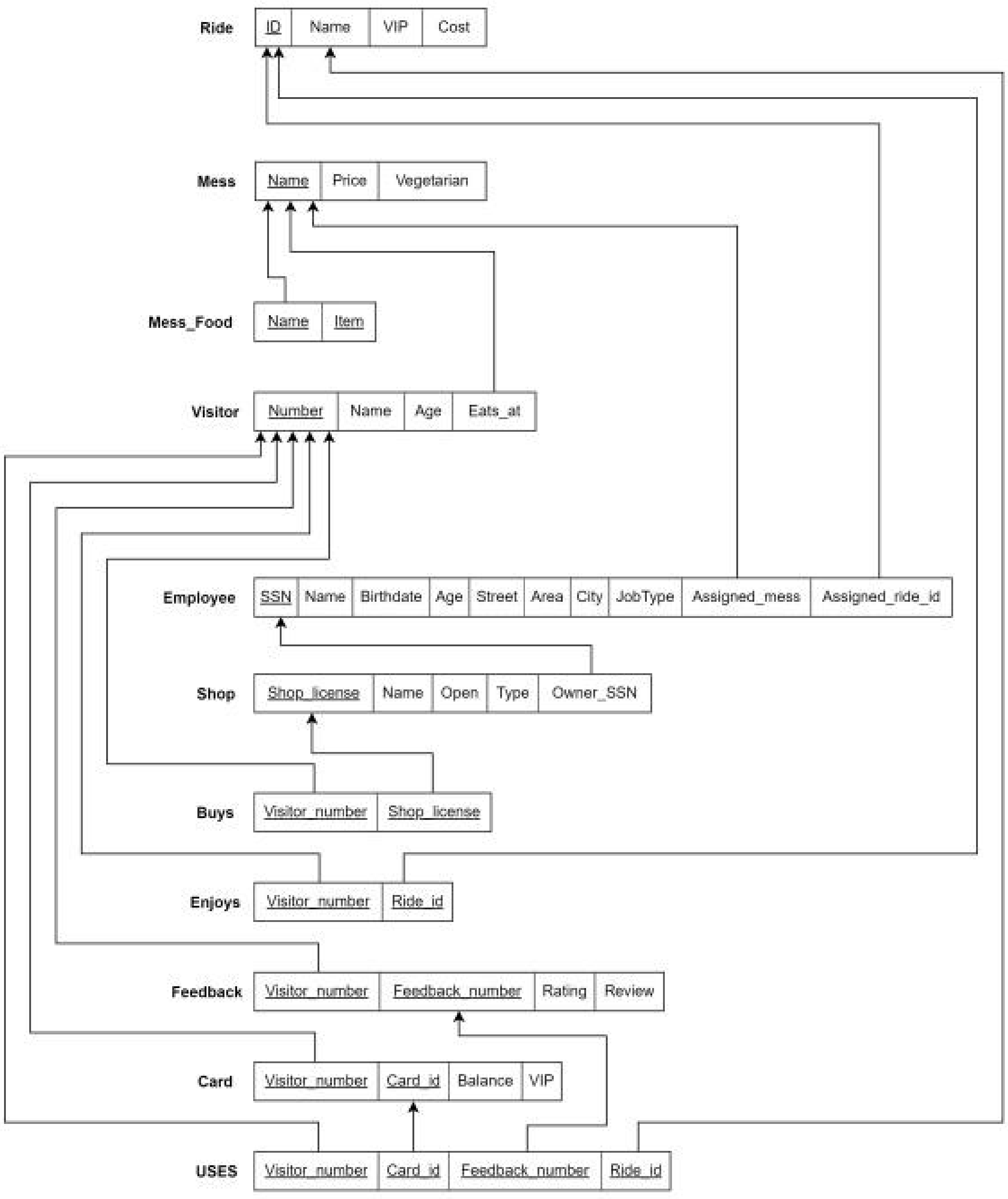

Ride
ID | Name | VIP | Cost
Mess
Name | Price | Vegetarian
Mess_Food
Name | Item
Visitor
Number | Name | Age | Eats_at
Employee
SSN | Name | Birthdate | Age | Street | Area | City | JobType | Assigned_mess | Assigned_ride_id
Shop
Shop_license | Name | Open | Type | Owner_SSN
Buys
Visitor_number | Shop_license
Enjoys
Visitor_number | Ride_id
Feedback
Visitor_number | Feedback_number | Rating | Review
Card
Visitor_number | Card_id | Balance | VIP
USES
Visitor_number | Card_id | Feedback_number | Ride_id

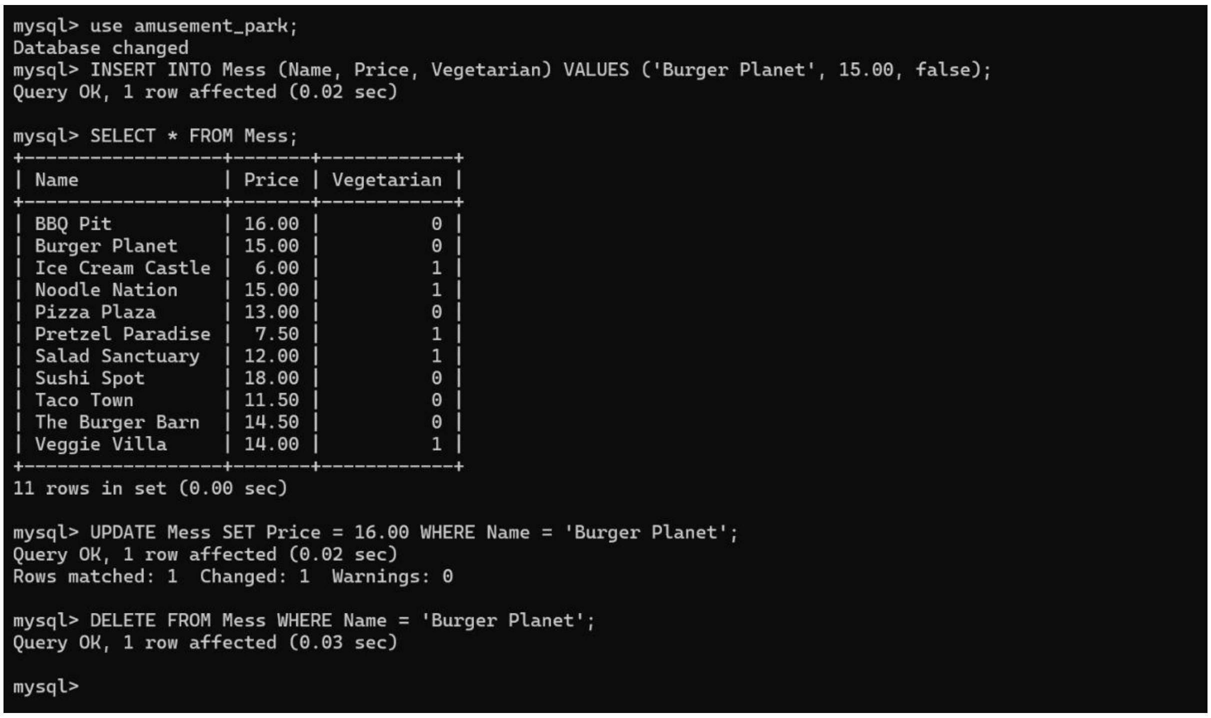
Figure 2: Relational Model

DDL COMMANDS:

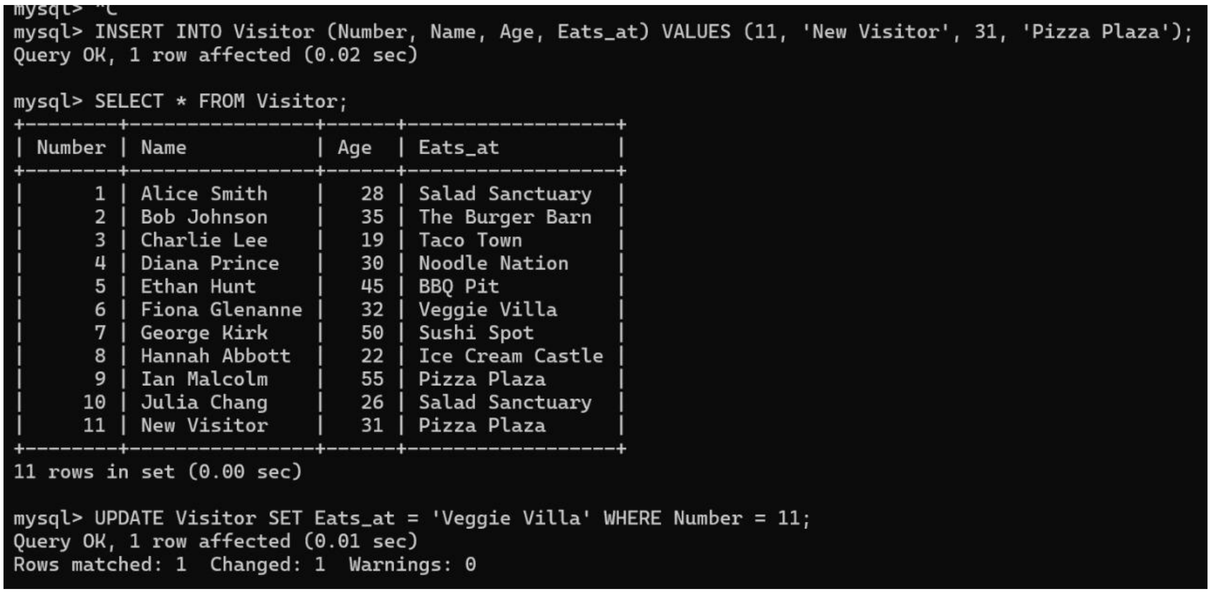
```
-- Select the database to use
USE amusement_park;

-- --------------------------------
-- PHASE 1: DDL (Data Definition Language)
-- Creating all 11 tables
-- --------------------------------

-- Table 1: Mess
CREATE TABLE Mess (
   Name VARCHAR(100) NOT NULL,
   Price DECIMAL(10, 2) NOT NULL,
   Vegetarian BOOLEAN DEFAULT false,
   PRIMARY KEY (Name)
);

-- Table 2: Ride
CREATE TABLE Ride (
   ID INT NOT NULL,
   Name VARCHAR(100) NOT NULL,
   VIP BOOLEAN DEFAULT false,
   Cost DECIMAL(10, 2) NOT NULL,
   PRIMARY KEY (ID)
);

-- Table 3: Visitor
CREATE TABLE Visitor (
   Number INT NOT NULL,
   Name VARCHAR(150) NOT NULL,
   Age INT CHECK (Age > 0),
   Eats_at VARCHAR(100) NULL,
   PRIMARY KEY (Number),
   FOREIGN KEY (Eats_at) REFERENCES Mess(Name)
       ON DELETE SET NULL
       ON UPDATE CASCADE
);

-- Table 4: Employee
CREATE TABLE Employee (
   SSN VARCHAR(20) NOT NULL,
   Name VARCHAR(150) NOT NULL,
   Birthdate DATE,
   Age INT,
   Street VARCHAR(100),
   Area VARCHAR(100),
   City VARCHAR(100),
   JobType VARCHAR(50),
   Assigned_mess VARCHAR(100) NULL,
   Assigned_ride_id INT NULL,
   PRIMARY KEY (SSN),
   FOREIGN KEY (Assigned_mess) REFERENCES Mess(Name)
       ON DELETE SET NULL
       ON UPDATE CASCADE,
   FOREIGN KEY (Assigned_ride_id) REFERENCES Ride(ID)
       ON DELETE SET NULL
       ON UPDATE CASCADE
);

-- Table 5: Shop
CREATE TABLE Shop (
   Shop_license VARCHAR(50) NOT NULL,
   Name VARCHAR(100),
   Open TIME,
   Type VARCHAR(50),
   Owner_SSN VARCHAR(20) NULL,
   PRIMARY KEY (Shop_license),
   FOREIGN KEY (Owner_SSN) REFERENCES Employee(SSN)
       ON DELETE SET NULL
);
```

```
-- Table 6: Mess_Food
CREATE TABLE Mess_Food (
    Name VARCHAR(100) NOT NULL,
    Item VARCHAR(100) NOT NULL,
    PRIMARY KEY (Name, Item),
    FOREIGN KEY (Name) REFERENCES Mess(Name)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

-- Table 7: Buys
CREATE TABLE Buys (
    Visitor_number INT NOT NULL,
    Shop_license VARCHAR(50) NOT NULL,
    PRIMARY KEY (Visitor_number, Shop_license),
    FOREIGN KEY (Visitor_number) REFERENCES Visitor(Number)
        ON DELETE CASCADE,
    FOREIGN KEY (Shop_license) REFERENCES Shop(Shop_license)
        ON DELETE CASCADE
);

-- Table 8: Enjoys
CREATE TABLE Enjoys (
    Visitor_number INT NOT NULL,
    Ride_id INT NOT NULL,
    PRIMARY KEY (Visitor_number, Ride_id),
    FOREIGN KEY (Visitor_number) REFERENCES Visitor(Number)
        ON DELETE CASCADE,
    FOREIGN KEY (Ride_id) REFERENCES Ride(ID)
        ON DELETE CASCADE
);

-- Table 9: Feedback
CREATE TABLE Feedback (
    Visitor_number INT NOT NULL,
    Feedback_number INT NOT NULL,
    Rating INT CHECK (Rating >= 1 AND Rating <= 5),
    Review TEXT,
    PRIMARY KEY (Visitor_number, Feedback_number),
    FOREIGN KEY (Visitor_number) REFERENCES Visitor(Number)
        ON DELETE CASCADE
);

-- Table 10: Card
CREATE TABLE Card (
    Visitor_number INT NOT NULL,
    Card_id VARCHAR(50) NOT NULL,
    Balance DECIMAL(10, 2) DEFAULT 0.00,
    VIP BOOLEAN DEFAULT false,
    PRIMARY KEY (Visitor_number, Card_id),
    FOREIGN KEY (Visitor_number) REFERENCES Visitor(Number)
        ON DELETE CASCADE
);

-- Table 11: USES
CREATE TABLE USES (
    Visitor_number INT NOT NULL,
    Card_id VARCHAR(50) NOT NULL,
    Feedback_number INT NOT NULL,
    Ride_id INT NOT NULL,
    PRIMARY KEY (Visitor_number, Card_id, Feedback_number,
Ride_id),

    FOREIGN KEY (Visitor_number, Card_id) REFERENCES
Card(Visitor_number, Card_id),
    FOREIGN KEY (Visitor_number, Feedback_number) REFERENCES
Feedback(Visitor_number, Feedback_number),
    FOREIGN KEY (Ride_id) REFERENCES Ride(ID)
);
```

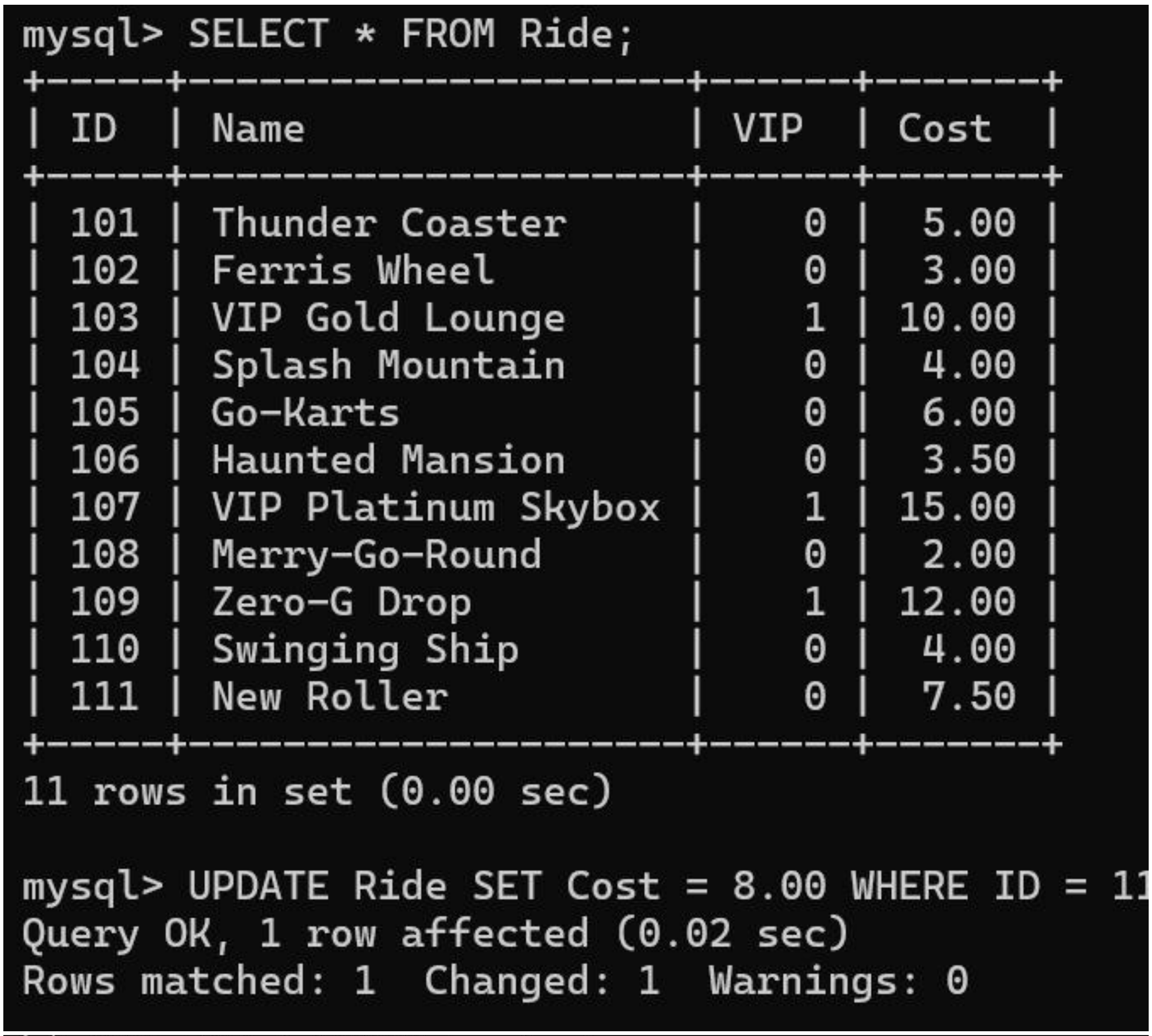
CRUD OPERATIONS:



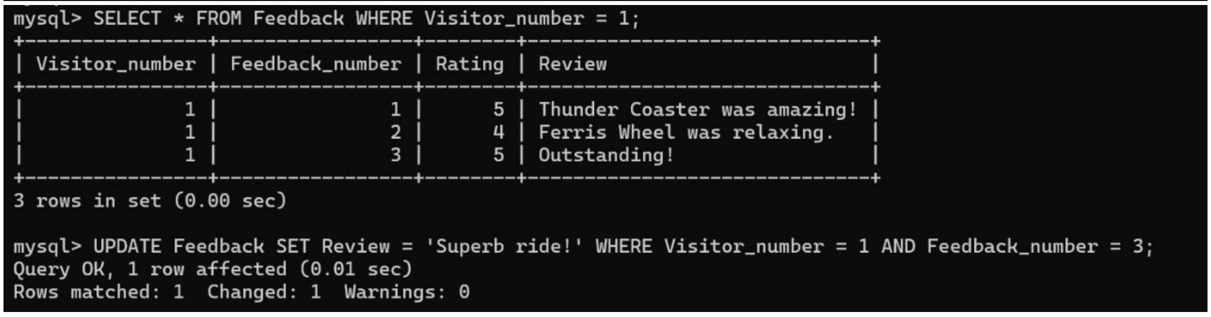
```
mysql> use amusement_park;
Database changed
mysql> INSERT INTO Mess (Name, Price, Vegetarian) VALUES ('Burger Planet', 15.00, false);
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM Mess;
+------------------+-------+------------+
| Name             | Price | Vegetarian |
+------------------+-------+------------+
| BBQ Pit          | 16.00 |          0 |
| Burger Planet    | 15.00 |          0 |
| Ice Cream Castle |  6.00 |          1 |
| Noodle Nation    | 15.00 |          1 |
| Pizza Plaza      | 13.00 |          0 |
| Pretzel Paradise |  7.50 |          1 |
| Salad Sanctuary  | 12.00 |          1 |
| Sushi Spot       | 18.00 |          0 |
| Taco Town        | 11.50 |          0 |
| The Burger Barn  | 14.50 |          0 |
| Veggie Villa     | 14.00 |          1 |
+------------------+-------+------------+
11 rows in set (0.00 sec)

mysql> UPDATE Mess SET Price = 16.00 WHERE Name = 'Burger Planet';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> DELETE FROM Mess WHERE Name = 'Burger Planet';
Query OK, 1 row affected (0.03 sec)

mysql>
```



```
mysql> ^C
mysql> INSERT INTO Visitor (Number, Name, Age, Eats_at) VALUES (11, 'New Visitor', 31, 'Pizza Plaza');
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM Visitor;
+--------+-----------------+------+------------------+
| Number | Name            | Age  | Eats_at          |
+--------+-----------------+------+------------------+
|      1 | Alice Smith     |   28 | Salad Sanctuary  |
|      2 | Bob Johnson     |   35 | The Burger Barn  |
|      3 | Charlie Lee     |   19 | Taco Town        |
|      4 | Diana Prince    |   30 | Noodle Nation    |
|      5 | Ethan Hunt      |   45 | BBQ Pit          |
|      6 | Fiona Glenanne  |   32 | Veggie Villa     |
|      7 | George Kirk     |   50 | Sushi Spot       |
|      8 | Hannah Abbott   |   22 | Ice Cream Castle |
|      9 | Ian Malcolm     |   55 | Pizza Plaza      |
|     10 | Julia Chang     |   26 | Salad Sanctuary  |
|     11 | New Visitor     |   31 | Pizza Plaza      |
+--------+-----------------+------+------------------+
11 rows in set (0.00 sec)

mysql> UPDATE Visitor SET Eats_at = 'Veggie Villa' WHERE Number = 11;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```



```
mysql> SELECT * FROM Ride;
+-----+---------------------+------+-------+
| ID  | Name                | VIP  | Cost  |
+-----+---------------------+------+-------+
| 101 | Thunder Coaster     |    0 |  5.00 |
| 102 | Ferris Wheel        |    0 |  3.00 |
| 103 | VIP Gold Lounge     |    1 | 10.00 |
| 104 | Splash Mountain     |    0 |  4.00 |
| 105 | Go-Karts            |    0 |  6.00 |
| 106 | Haunted Mansion     |    0 |  3.50 |
| 107 | VIP Platinum Skybox |    1 | 15.00 |
| 108 | Merry-Go-Round      |    0 |  2.00 |
| 109 | Zero-G Drop         |    1 | 12.00 |
| 110 | Swinging Ship       |    0 |  4.00 |
| 111 | New Roller          |    0 |  7.50 |
+-----+---------------------+------+-------+
11 rows in set (0.00 sec)

mysql> UPDATE Ride SET Cost = 8.00 WHERE ID = 11
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```



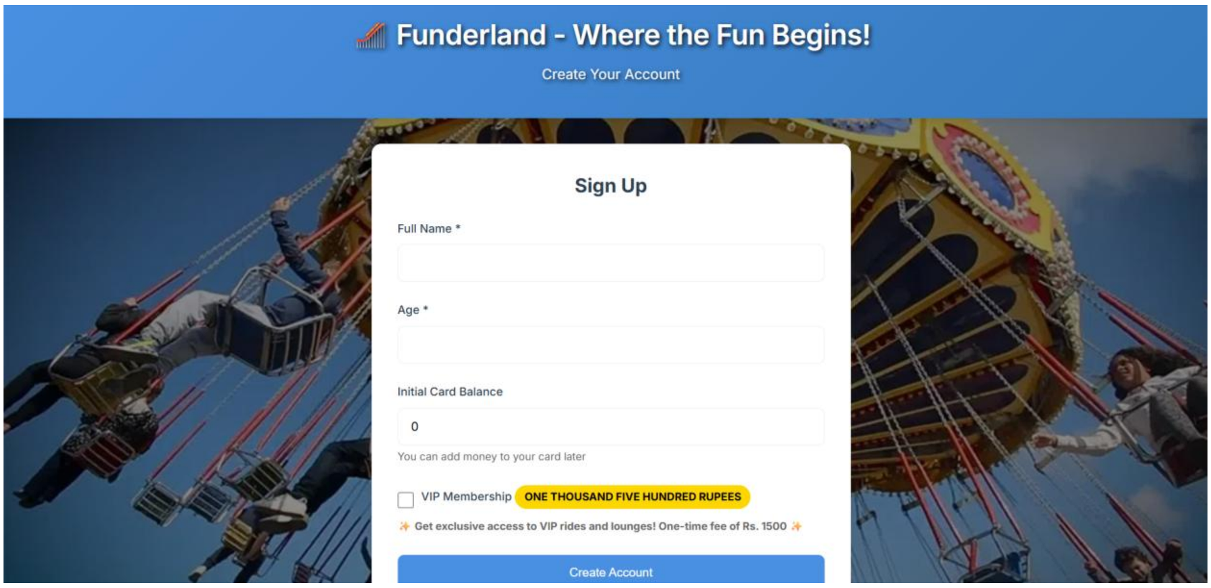
```
mysql> SELECT * FROM Feedback WHERE Visitor_number = 1;
+----------------+-----------------+--------+-----------------------------+
| Visitor_number | Feedback_number | Rating | Review                      |
+----------------+-----------------+--------+-----------------------------+
|              1 |               1 |      5 | Thunder Coaster was amazing! |
|              1 |               2 |      4 | Ferris Wheel was relaxing.  |
|              1 |               3 |      5 | Outstanding!                |
+----------------+-----------------+--------+-----------------------------+
3 rows in set (0.00 sec)

mysql> UPDATE Feedback SET Review = 'Superb ride!' WHERE Visitor_number = 1 AND Feedback_number = 3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

LIST OF FUNCTIONALITIES/FEATURES OF THE APPLICATION:

1. User Registration/Sign Up

- **Description: New visitors can create an account by entering their name, email, password, and age**

- **Database Operation: INSERT into Visitor and Card tables**



Funderland - Where the Fun Begins!

Create Your Account

Sign Up

Full Name *

Age *

Initial Card Balance

0

You can add money to your card later

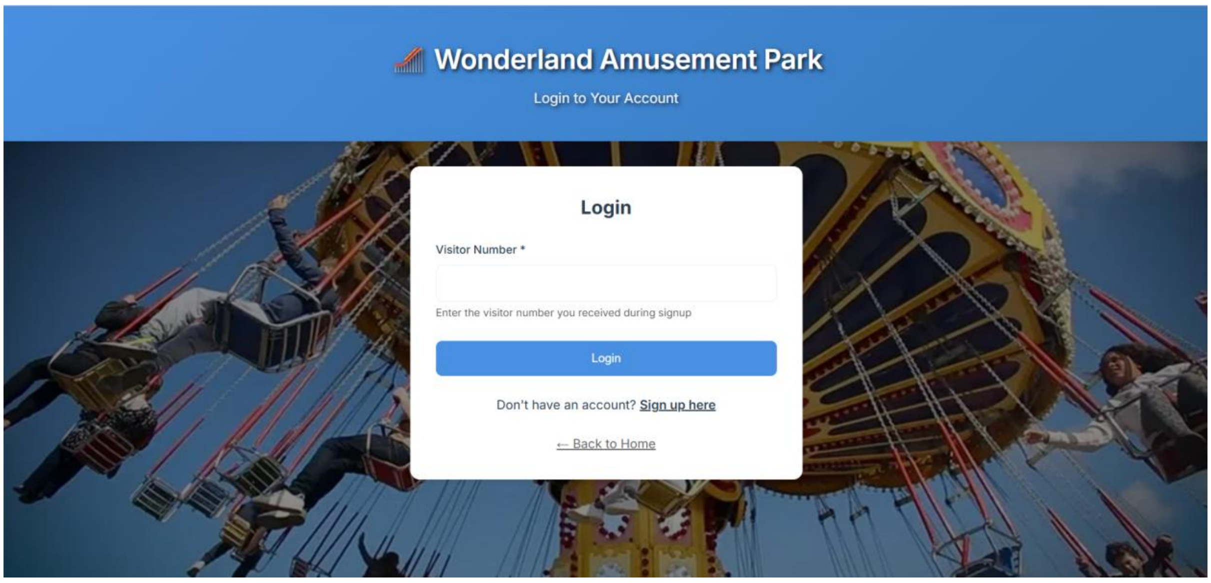
VIP Membership ONE THOUSAND FIVE HUNDRED RUPEES

✨ Get exclusive access to VIP rides and lounges! One-time fee of Rs. 1500 ✨

Create Account

2. User Login/Authentication

- **Description: Registered users can log in using email and password to access personalized features**
- **Database Operation: SELECT from Visitor/User table with password verification**



Wonderland Amusement Park

Login to Your Account

Login

Visitor Number *

Enter the visitor number you received during signup

Login

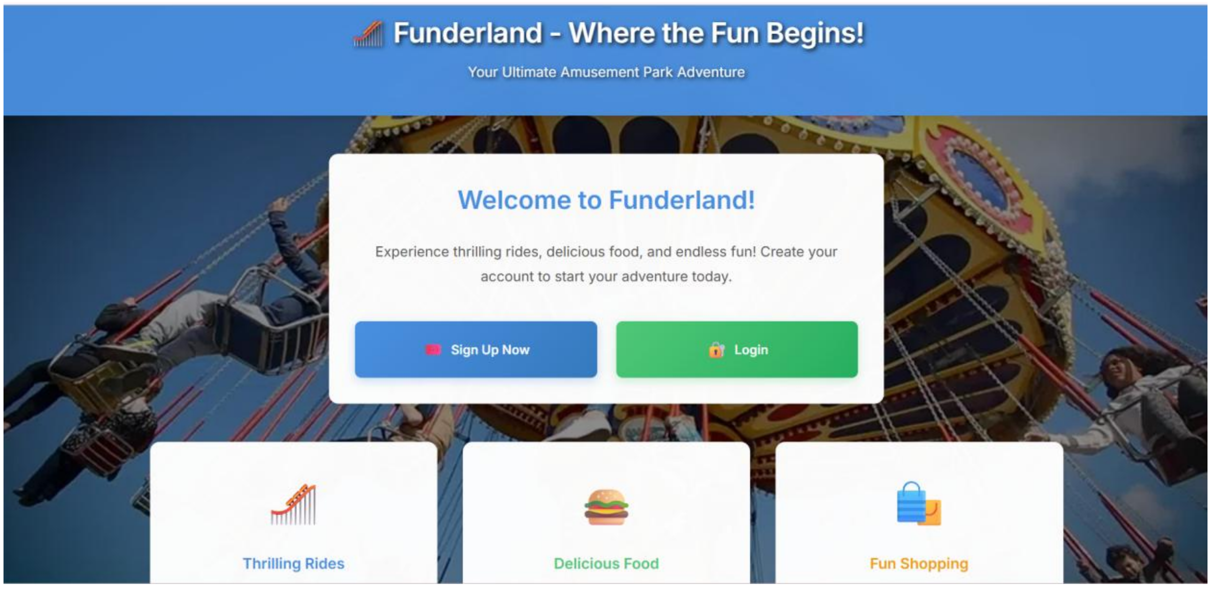
Don't have an account? Sign up here

← Back to Home

3. User Dashboard/Home Page

- **Description: After login, users see a personalized dashboard with navigation to all features**

- **Features visible: Links to Rides, Mess/Food, Shop, Bookings, Feedback sections**

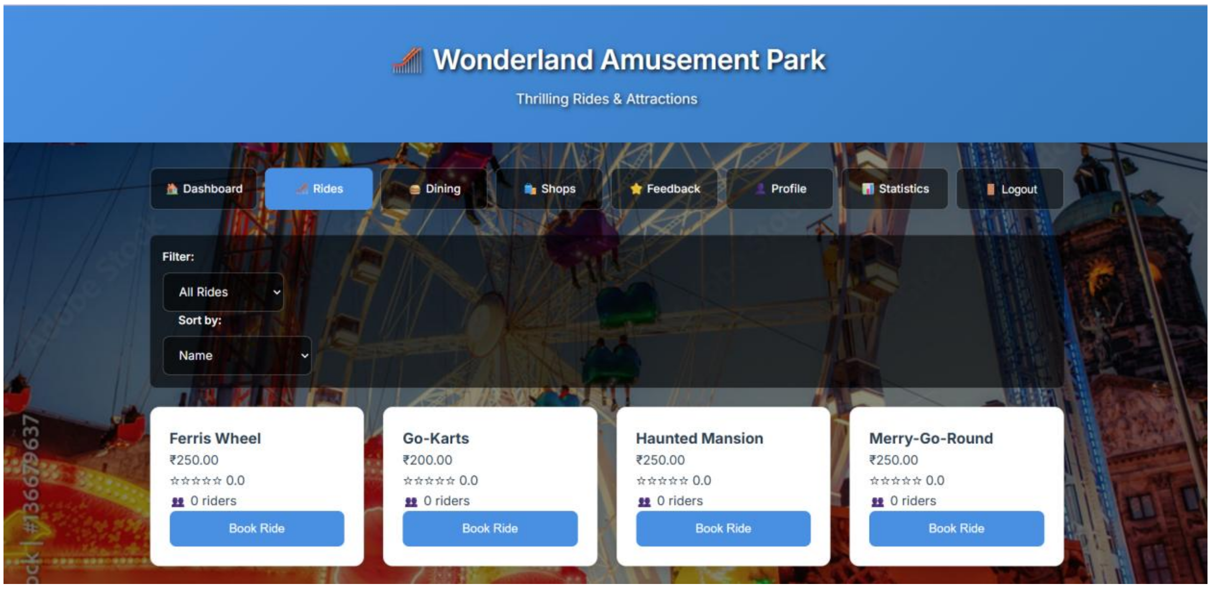

Funderland - Where the Fun Begins!
Your Ultimate Amusement Park Adventure
Sign Up Now
Login
Thrilling Rides
Delicious Food
Fun Shopping

Welcome to Funderland!

Experience thrilling rides, delicious food, and endless fun! Create your account to start your adventure today.

4. Browse Available Rides

- **Description: Users can view all available rides with details (name, cost, VIP status)**
- **Database Operation: SELECT from Ride table**
- **Display fields: Ride name, cost, VIP badge if applicable**



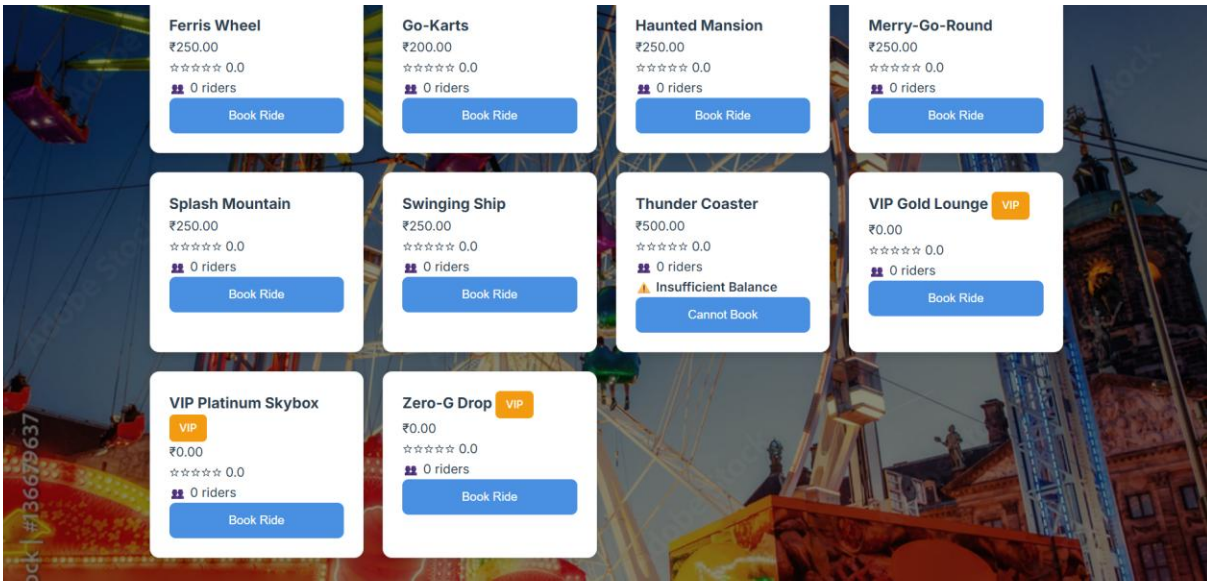
Filter: All Rides
Sort by: Name

Ferris Wheel
₹250.00
☆☆☆☆☆ 0.0
0 riders

Go-Karts
₹200.00
☆☆☆☆☆ 0.0
0 riders

Haunted Mansion
₹250.00
☆☆☆☆☆ 0.0
0 riders

Merry-Go-Round
₹250.00
☆☆☆☆☆ 0.0
0 riders


Ferris Wheel
₹250.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Go-Karts
₹200.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Haunted Mansion
₹250.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Merry-Go-Round
₹250.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Splash Mountain
₹250.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Swinging Ship
₹250.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Thunder Coaster
₹500.00
☆☆☆☆☆ 0.0
0 riders
Insufficient Balance
Cannot Book

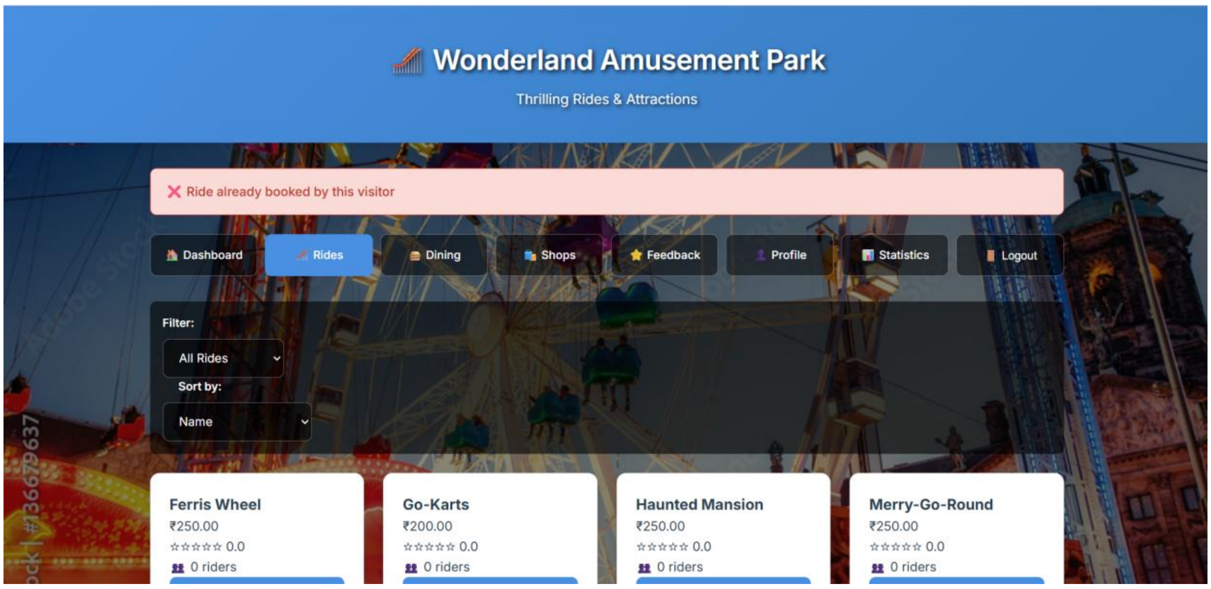
VIP Gold Lounge VIP
₹0.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

VIP Platinum Skybox VIP
₹0.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

Zero-G Drop VIP
₹0.00
☆☆☆☆☆ 0.0
0 riders
Book Ride

5. Book/Enjoy a Ride

- **Description: Users can select and book rides (VIP verification applied automatically)**
- **Database Operation: INSERT into Enjoys table (trigger checks VIP status)**
- **Success/Error messages: "Booking successful!" or "VIP access required"**


Wonderland Amusement Park
Thrilling Rides & Attractions

✕ Ride already booked by this visitor

Filter: All Rides
Sort by: Name

Ferris Wheel
₹250.00
☆☆☆☆☆ 0.0
0 riders

Go-Karts
₹200.00
☆☆☆☆☆ 0.0
0 riders

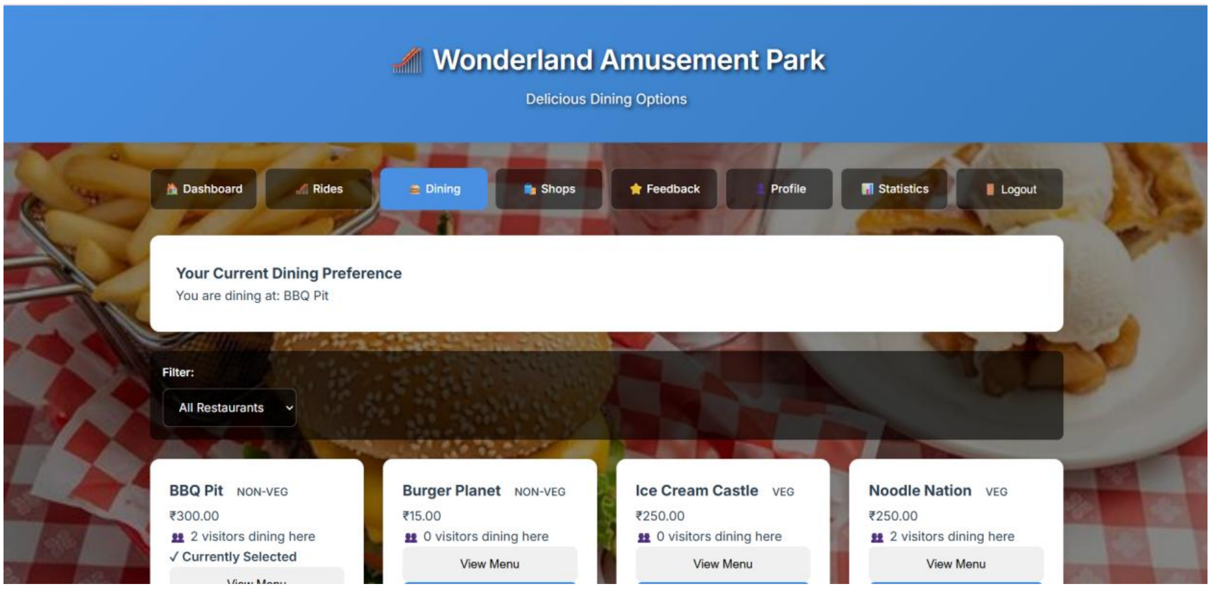
Haunted Mansion
₹250.00
☆☆☆☆☆ 0.0
0 riders

Merry-Go-Round
₹250.00
☆☆☆☆☆ 0.0
0 riders

6. View Mess/Food Items

- **Description: Browse available food outlets with prices and vegetarian options**

- **Database Operation: SELECT from Mess and Mess_Food tables with JOIN**
- **Display fields: Mess name, price, vegetarian indicator, food items**



Wonderland Amusement Park

Delicious Dining Options

Your Current Dining Preference

You are dining at: BBQ Pit

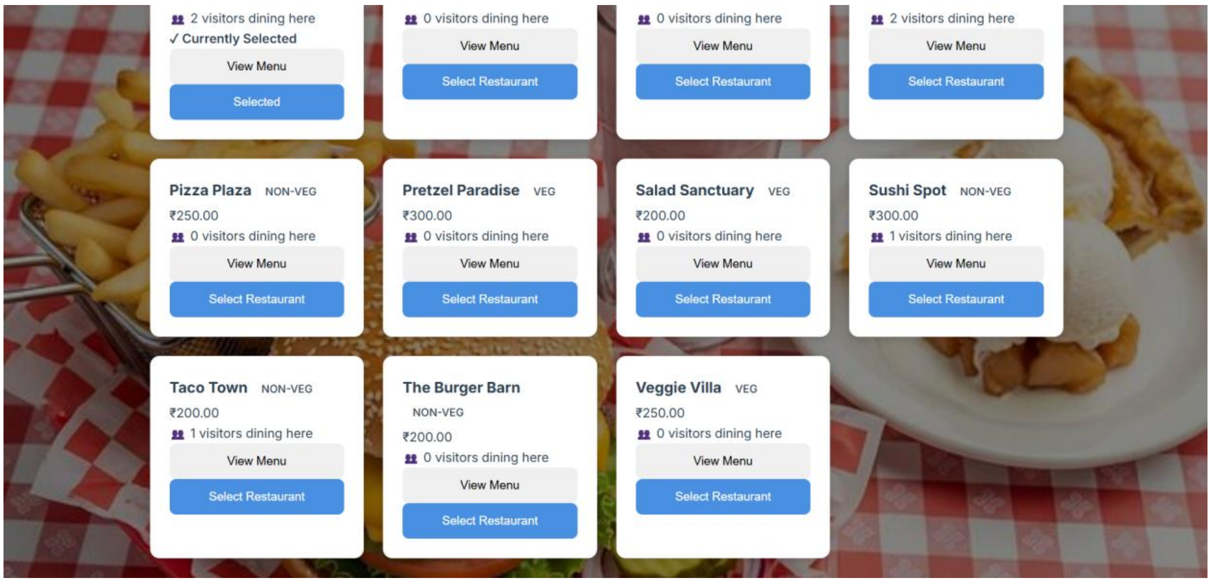
Filter: All Restaurants

BBQ Pit NON-VEG
₹300.00
2 visitors dining here
✓ Currently Selected

Burger Planet NON-VEG
₹15.00
0 visitors dining here
View Menu

Ice Cream Castle VEG
₹250.00
0 visitors dining here
View Menu

Noodle Nation VEG
₹250.00
2 visitors dining here
View Menu



2 visitors dining here
✓ Currently Selected
View Menu
Selected

0 visitors dining here
View Menu
Select Restaurant

0 visitors dining here
View Menu
Select Restaurant

2 visitors dining here
View Menu
Select Restaurant

Pizza Plaza NON-VEG
₹250.00
0 visitors dining here
View Menu
Select Restaurant

Pretzel Paradise VEG
₹300.00
0 visitors dining here
View Menu
Select Restaurant

Salad Sanctuary VEG
₹200.00
0 visitors dining here
View Menu
Select Restaurant

Sushi Spot NON-VEG
₹300.00
1 visitors dining here
View Menu
Select Restaurant

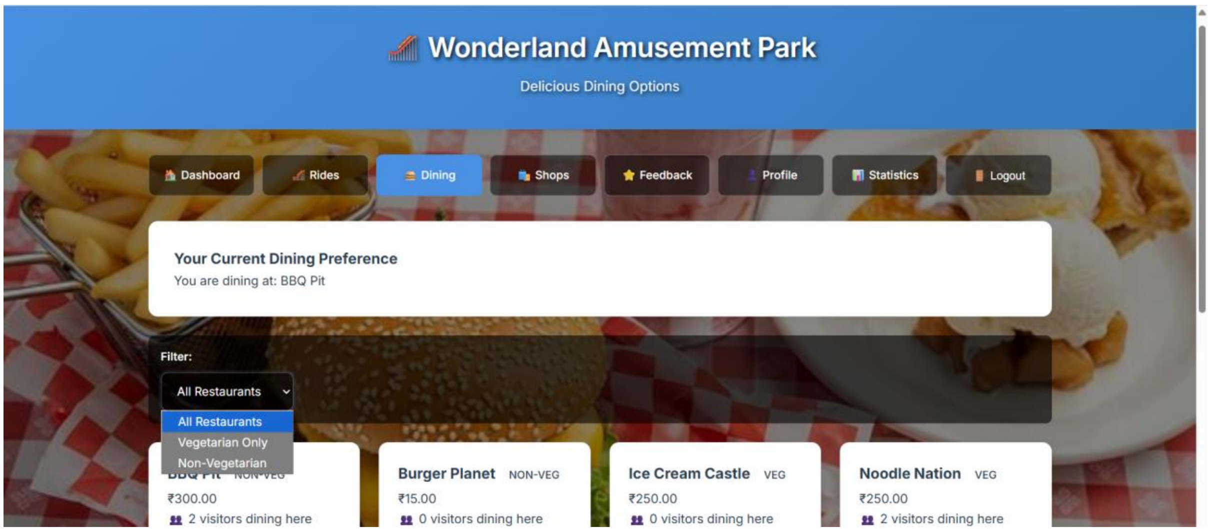
Taco Town NON-VEG
₹200.00
1 visitors dining here
View Menu
Select Restaurant

The Burger Barn NON-VEG
₹200.00
0 visitors dining here
View Menu
Select Restaurant

Veggie Villa VEG
₹250.00
0 visitors dining here
View Menu
Select Restaurant

7. Select Eating Location

- **Description: Users can choose where they want to eat**
- **Database Operation: UPDATE Visitor table (Eats_at field)**



Wonderland Amusement Park

Delicious Dining Options

Your Current Dining Preference

You are dining at: BBQ Pit

Filter:

All Restaurants

BBQ Pit NON-VEG
₹300.00
2 visitors dining here

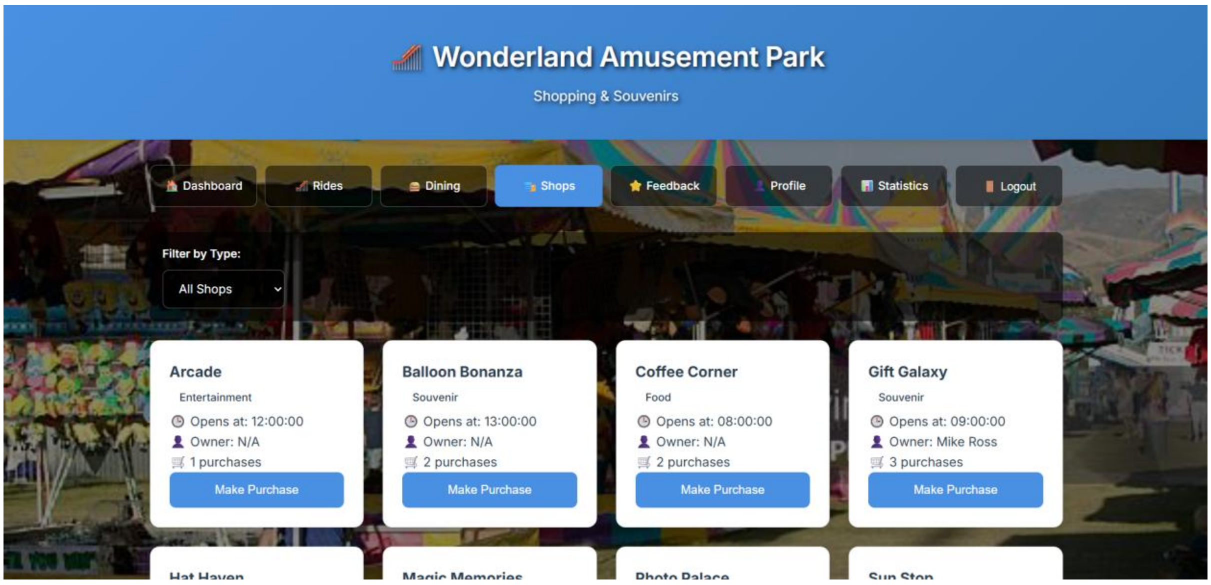
Burger Planet NON-VEG
₹15.00
0 visitors dining here

Ice Cream Castle VEG
₹250.00
0 visitors dining here

Noodle Nation VEG
₹250.00
2 visitors dining here

8. Browse Shop Items

- **Description: View available shops with their operating hours and types**
- **Database Operation: SELECT from Shop table**
- **Display fields: Shop name, type, opening time**



Wonderland Amusement Park

Shopping & Souvenirs

Filter by Type:

All Shops

Arcade
Entertainment
Opens at: 12:00:00
Owner: N/A
1 purchases
Make Purchase

Balloon Bonanza
Souvenir
Opens at: 13:00:00
Owner: N/A
2 purchases
Make Purchase

Coffee Corner
Food
Opens at: 08:00:00
Owner: N/A
2 purchases
Make Purchase

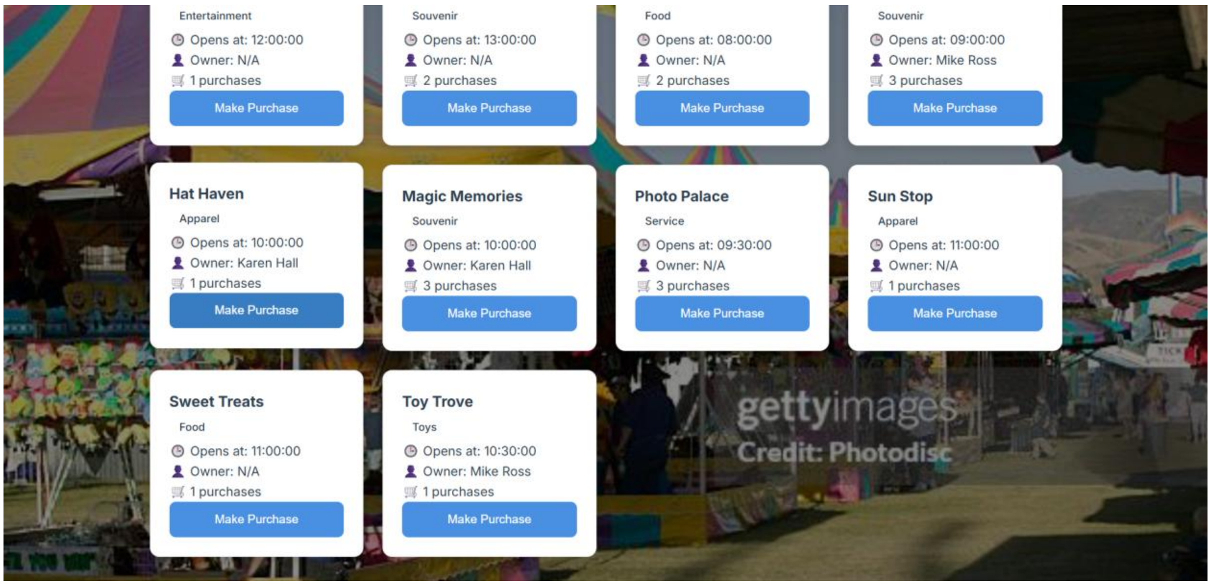
Gift Galaxy
Souvenir
Opens at: 09:00:00
Owner: Mike Ross
3 purchases
Make Purchase

Hat Haven

Magic Memories

Photo Palace

Sun Stop


Entertainment
Opens at: 12:00:00
Owner: N/A
1 purchases
Make Purchase

Souvenir
Opens at: 13:00:00
Owner: N/A
2 purchases
Make Purchase

Food
Opens at: 08:00:00
Owner: N/A
2 purchases
Make Purchase

Souvenir
Opens at: 09:00:00
Owner: Mike Ross
3 purchases
Make Purchase

Hat Haven
Apparel
Opens at: 10:00:00
Owner: Karen Hall
1 purchases
Make Purchase

Magic Memories
Souvenir
Opens at: 10:00:00
Owner: Karen Hall
3 purchases
Make Purchase

Photo Palace
Service
Opens at: 09:30:00
Owner: N/A
3 purchases
Make Purchase

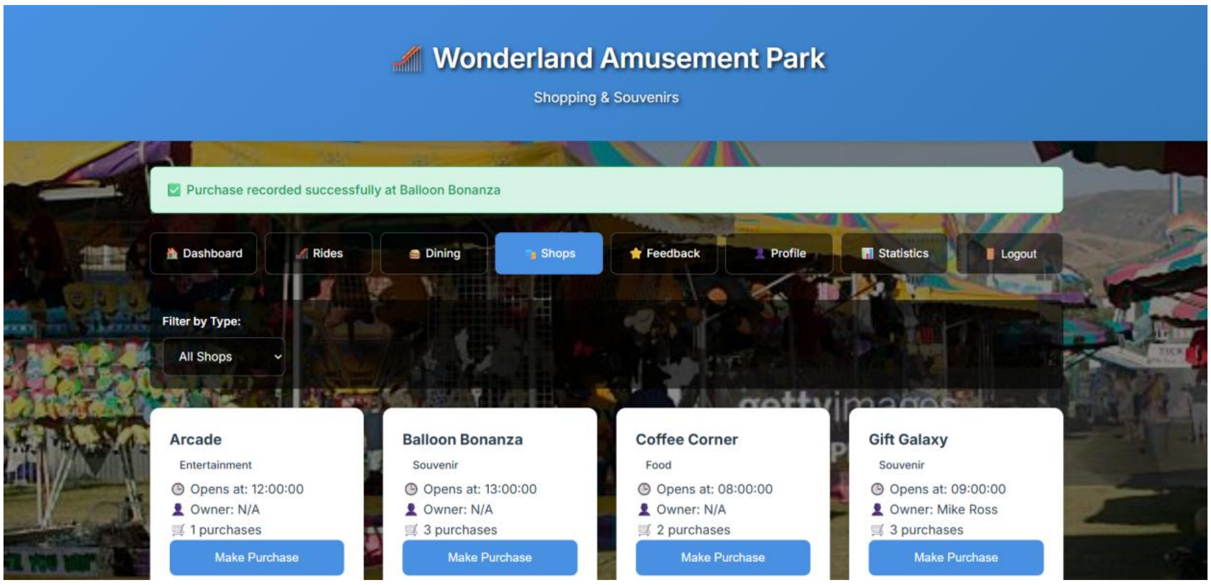
Sun Stop
Apparel
Opens at: 11:00:00
Owner: N/A
1 purchases
Make Purchase

Sweet Treats
Food
Opens at: 11:00:00
Owner: N/A
1 purchases
Make Purchase

Toy Trove
Toys
Opens at: 10:30:00
Owner: Mike Ross
1 purchases
Make Purchase

9. Make Purchases at Shops

- **Description: Users can purchase items from shops**
- **Database Operation: INSERT into Buys table**


Wonderland Amusement Park
Shopping & Souvenirs

Purchase recorded successfully at Balloon Bonanza

Filter by Type:
All Shops

Arcade
Entertainment
Opens at: 12:00:00
Owner: N/A
1 purchases
Make Purchase

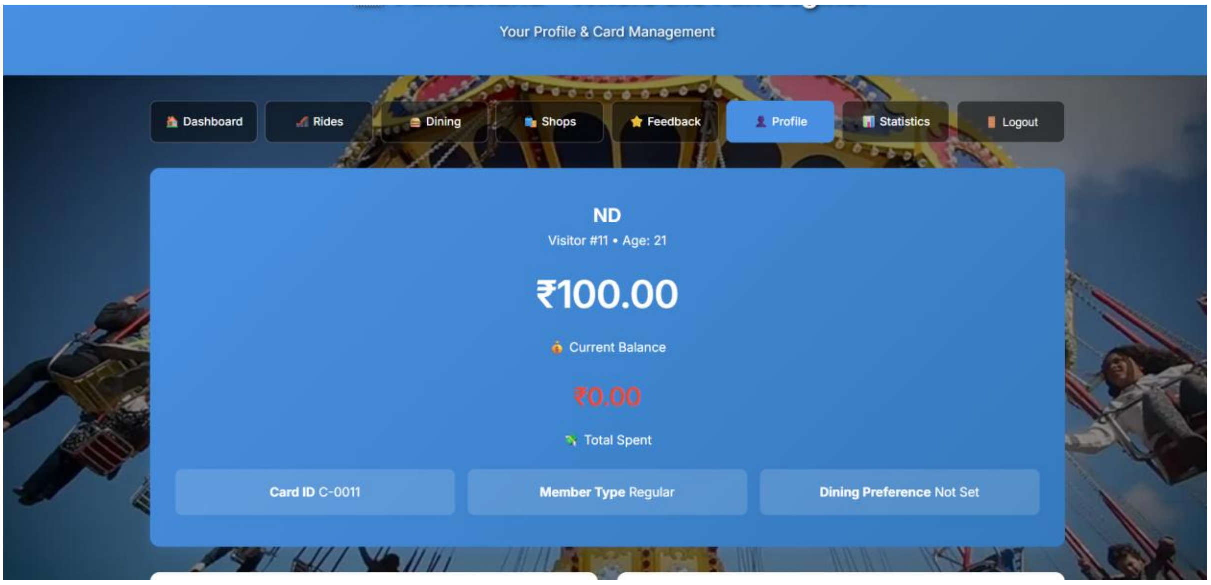
Balloon Bonanza
Souvenir
Opens at: 13:00:00
Owner: N/A
3 purchases
Make Purchase

Coffee Corner
Food
Opens at: 08:00:00
Owner: N/A
2 purchases
Make Purchase

Gift Galaxy
Souvenir
Opens at: 09:00:00
Owner: Mike Ross
3 purchases
Make Purchase

10. View Card Balance and Details

- **Description: Users can check their park card balance and VIP status**
- **Database Operation: SELECT from Card table**
- **Display fields: Card ID, current balance, VIP status indicator**


Your Profile & Card Management

ND

Visitor #11 • Age: 21

₹100.00

Current Balance

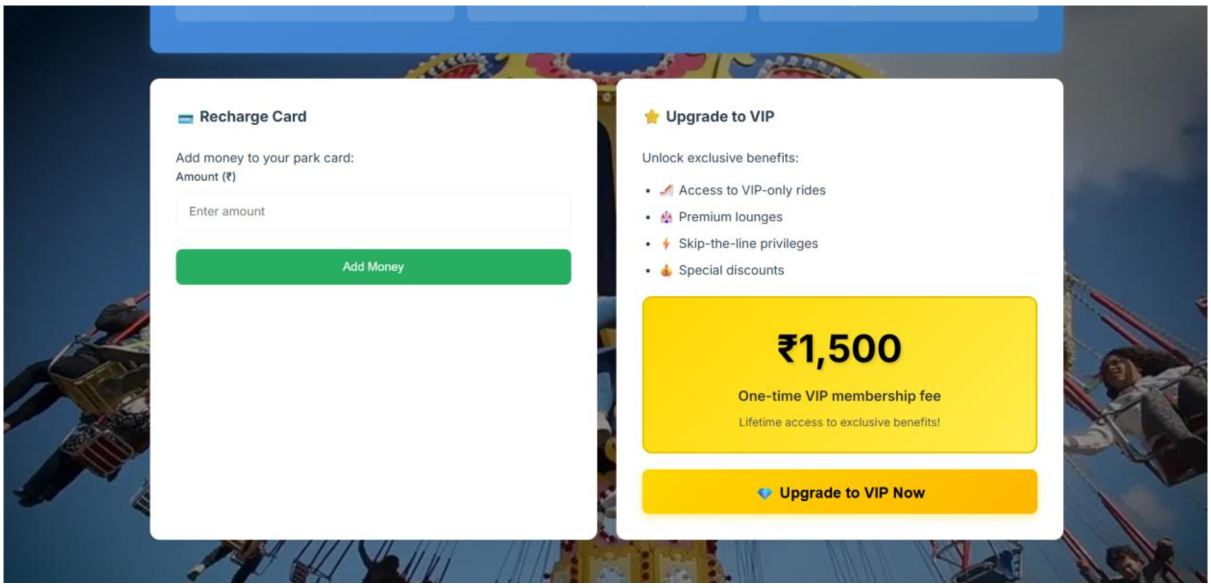
₹0.00

Total Spent

Card ID C-0011

Member Type Regular

Dining Preference Not Set


Recharge Card

Add money to your park card:

Amount (₹)

Enter amount

Upgrade to VIP

Unlock exclusive benefits:

- Access to VIP-only rides
- Premium lounges
- Skip-the-line privileges
- Special discounts

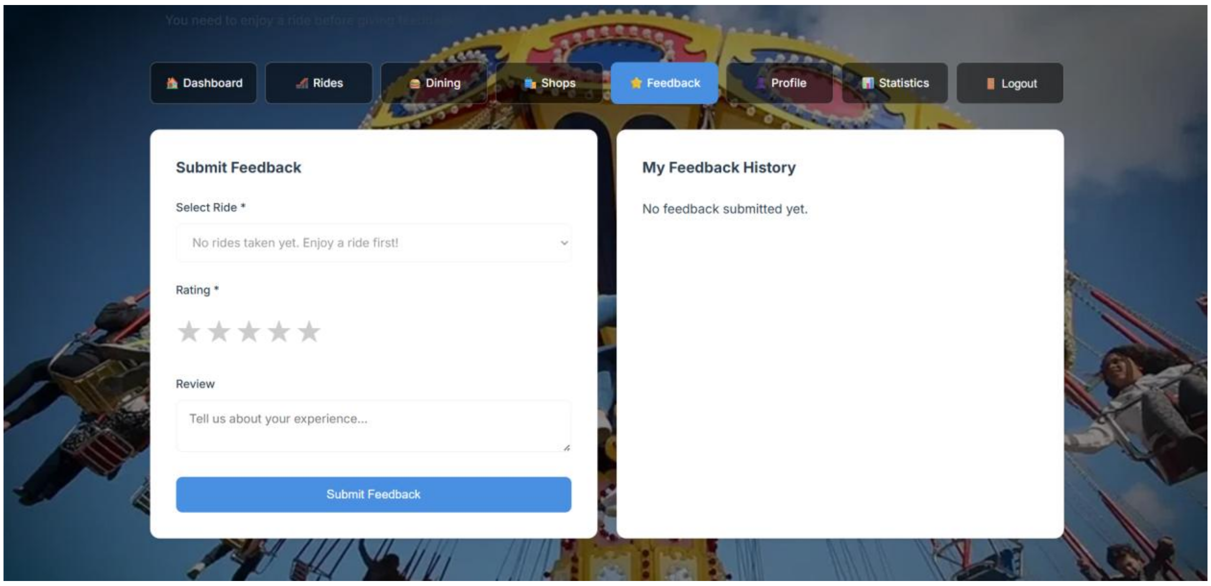
₹1,500

One-time VIP membership fee

Lifetime access to exclusive benefits!

11. Submit Feedback

- **Description: Users can rate their experience and write reviews**
- **Database Operation: INSERT into Feedback table**
- **Form fields: Rating dropdown/stars, review text area, submit button**


You need to enjoy a ride before giving feedback.

Submit Feedback

Select Ride *

No rides taken yet. Enjoy a ride first!

Rating *

Review

Tell us about your experience...

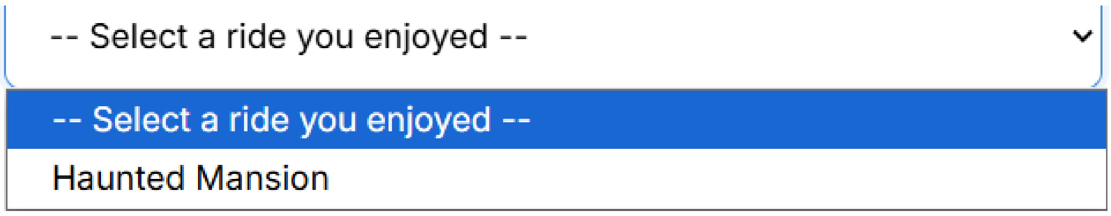
Submit Feedback

My Feedback History

No feedback submitted yet.

12. View My Bookings/Ride History

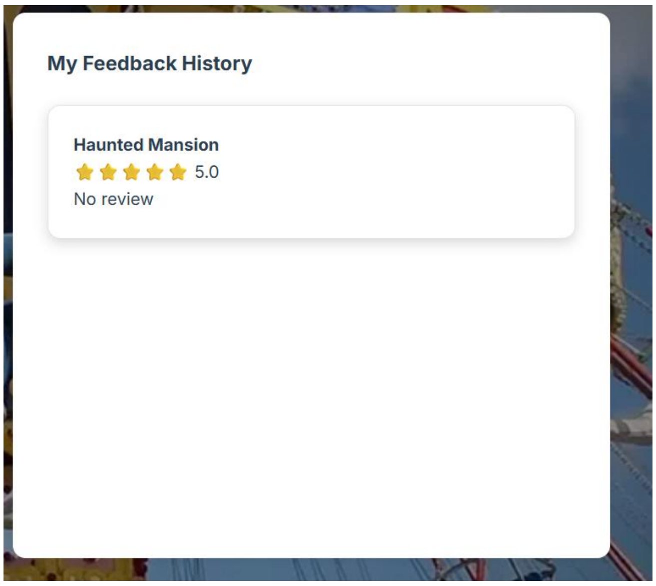
- **Description: Users can see all rides they have enjoyed**
- **Database Operation: SELECT from Enjoys JOIN Ride**
- **Display fields: Ride name, date (if tracked)**


-- Select a ride you enjoyed --

-- Select a ride you enjoyed --
Haunted Mansion

13. View My Feedback History

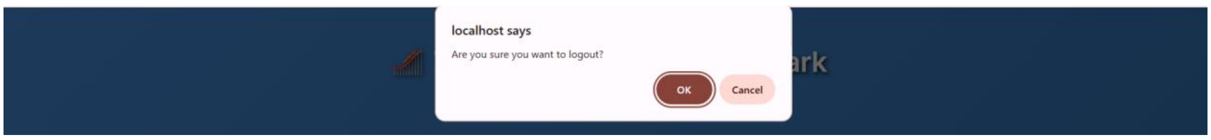
- **Description: Users can review feedback they've submitted**
- **Database Operation: SELECT from Feedback table filtered by user**


My Feedback History

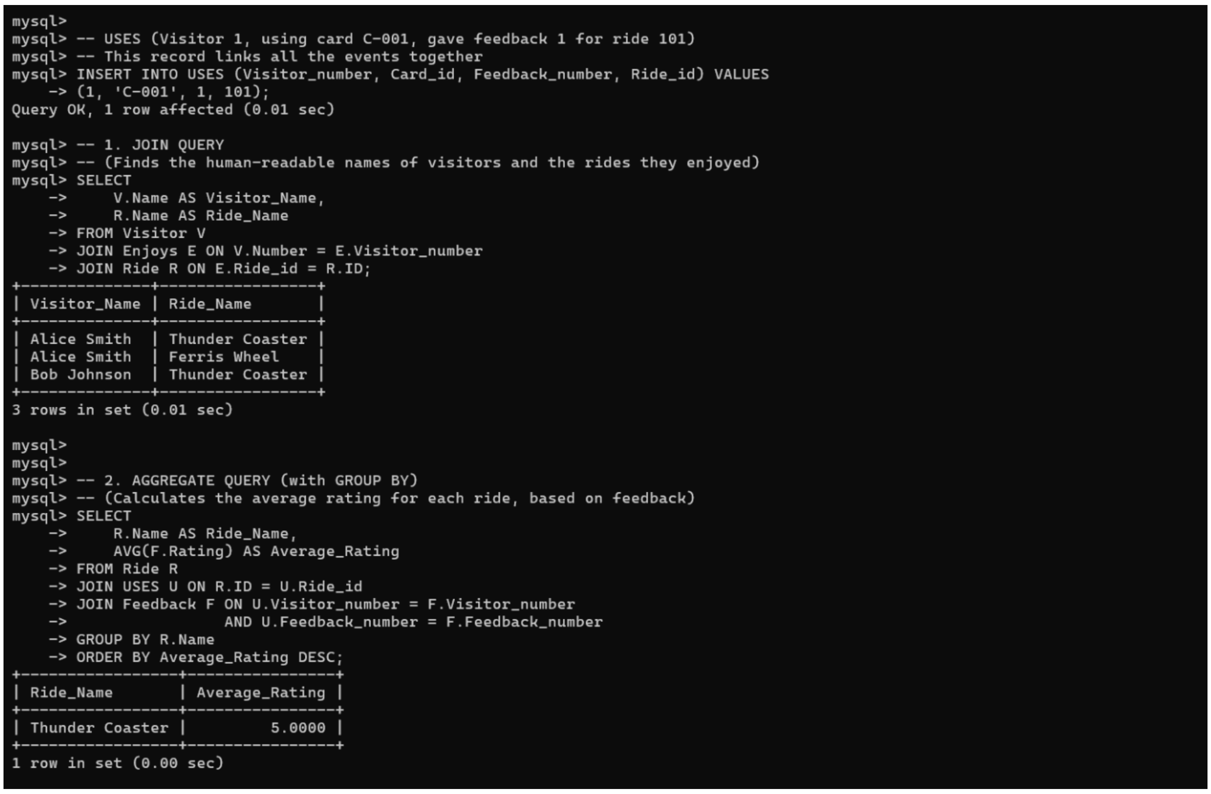
Haunted Mansion
★★★★★ 5.0
No review

14. Logout Functionality

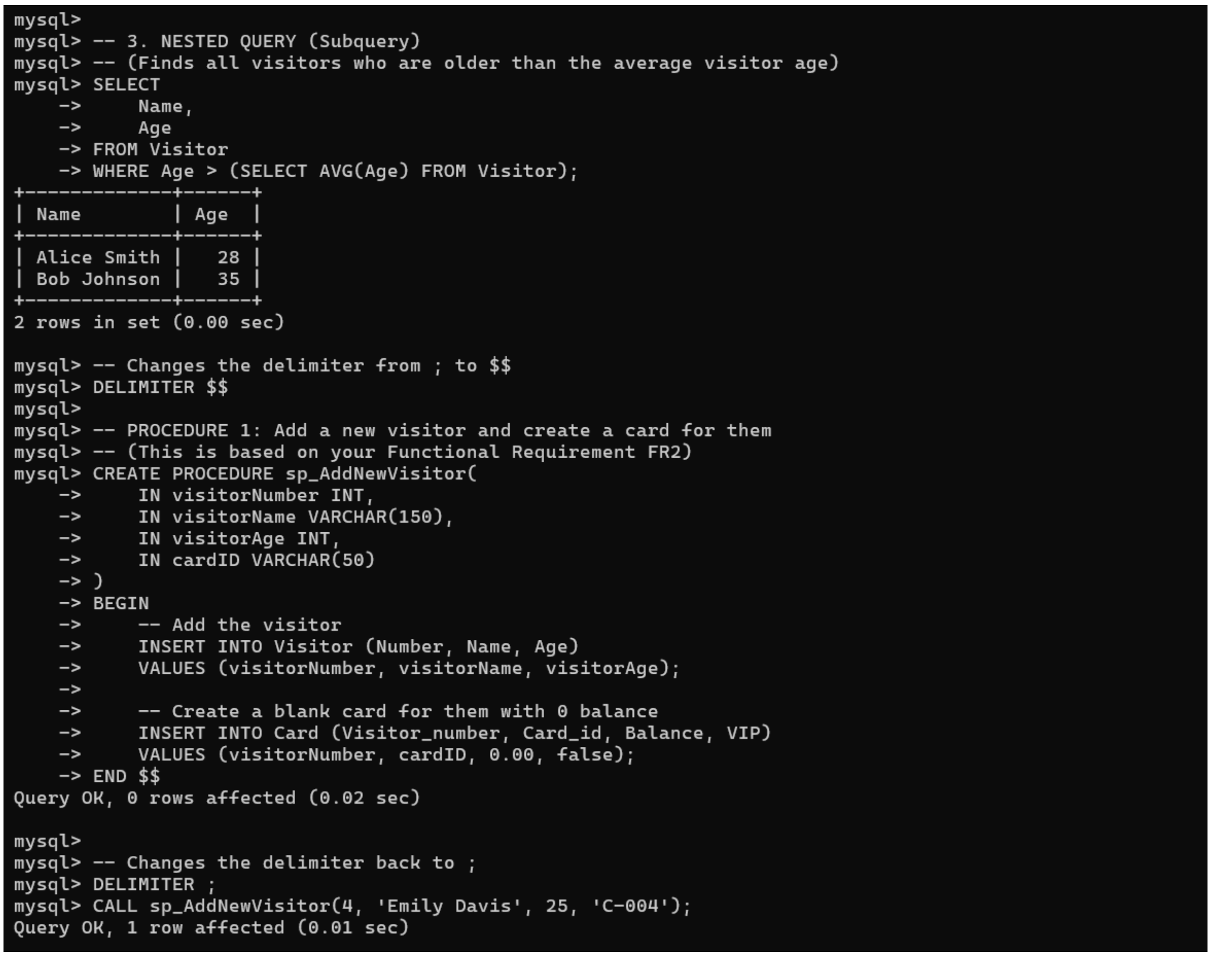
- **Description: Users can securely log out of their session**
- **Action: Session termination, redirect to login page**


localhost says
Are you sure you want to logout?

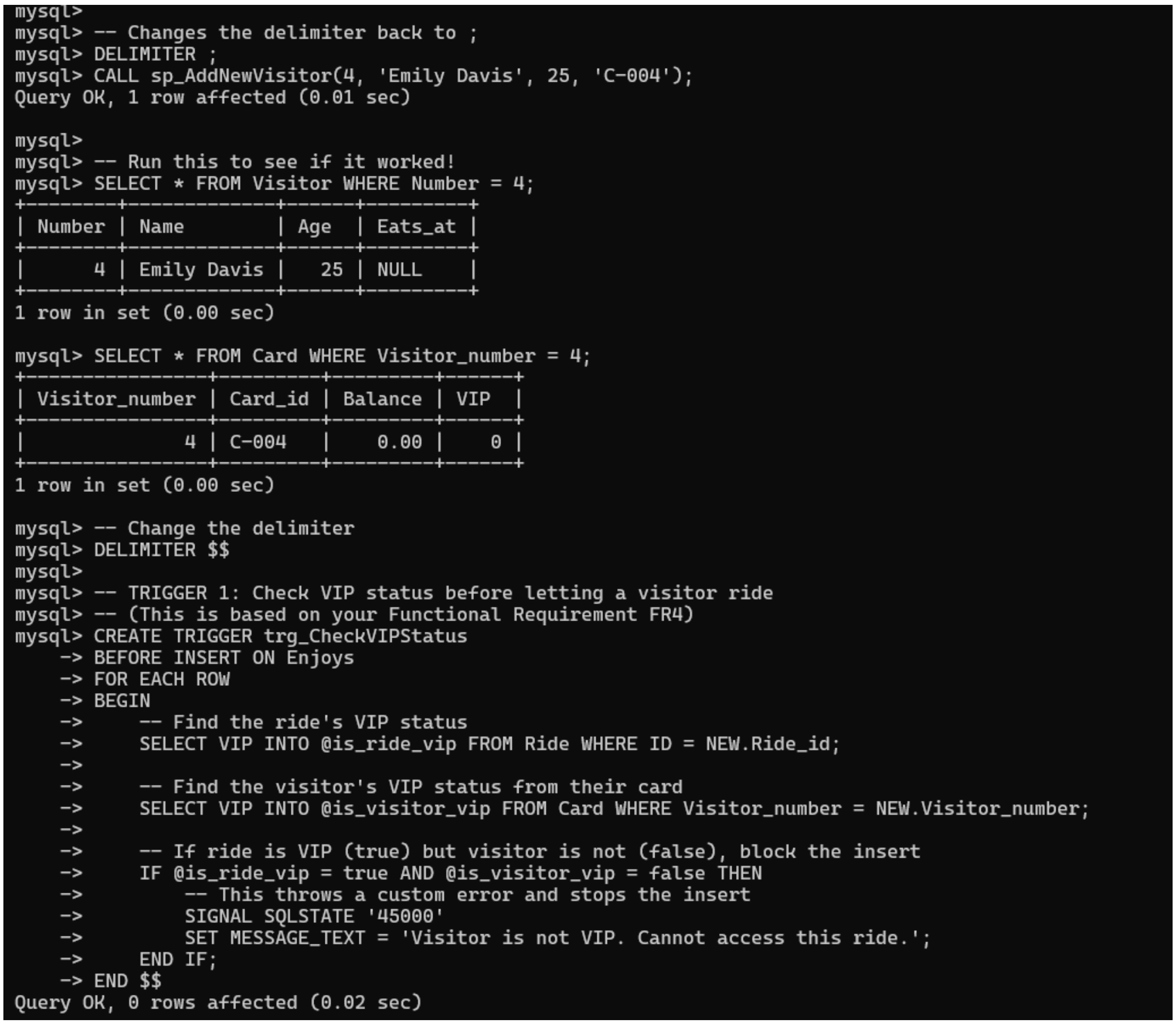
TRIGGERS, PROCEDURES/FUNCTIONS, NESTED QUERY, JOIN, AGGREGATE QUERIES



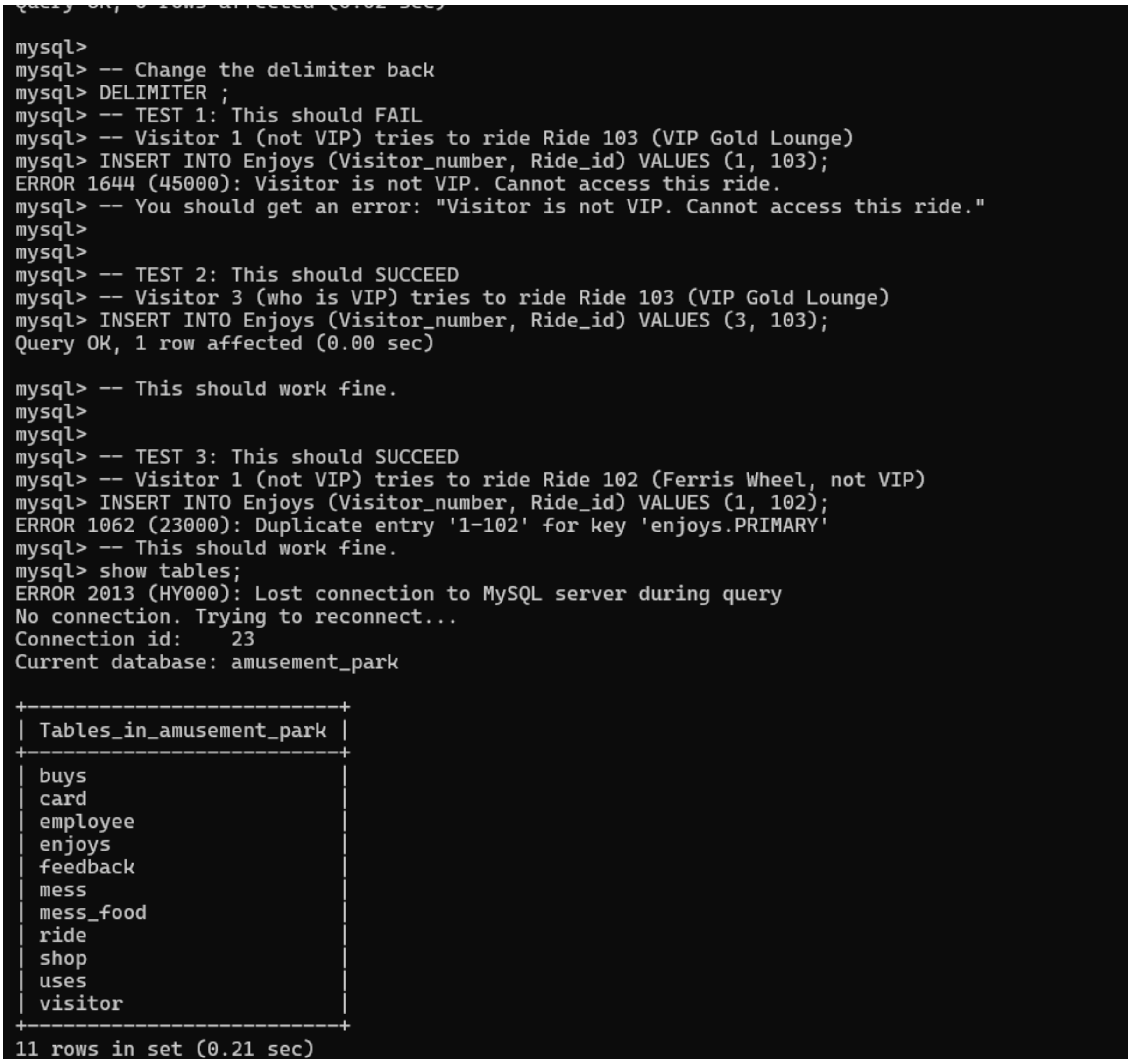
```
mysql>
mysql> -- USES (Visitor 1, using card C-001, gave feedback 1 for ride 101)
mysql> -- This record links all the events together
mysql> INSERT INTO USES (Visitor_number, Card_id, Feedback_number, Ride_id) VALUES
    -> (1, 'C-001', 1, 101);
Query OK, 1 row affected (0.01 sec)

mysql> -- 1. JOIN QUERY
mysql> -- (Finds the human-readable names of visitors and the rides they enjoyed)
mysql> SELECT
    ->     V.Name AS Visitor_Name,
    ->     R.Name AS Ride_Name
    -> FROM Visitor V
    -> JOIN Enjoys E ON V.Number = E.Visitor_number
    -> JOIN Ride R ON E.Ride_id = R.ID;
+--------------+----------------+
| Visitor_Name | Ride_Name      |
+--------------+----------------+
| Alice Smith  | Thunder Coaster |
| Alice Smith  | Ferris Wheel   |
| Bob Johnson  | Thunder Coaster |
+--------------+----------------+
3 rows in set (0.01 sec)

mysql>
mysql>
mysql> -- 2. AGGREGATE QUERY (with GROUP BY)
mysql> -- (Calculates the average rating for each ride, based on feedback)
mysql> SELECT
    ->     R.Name AS Ride_Name,
    ->     AVG(F.Rating) AS Average_Rating
    -> FROM Ride R
    -> JOIN USES U ON R.ID = U.Ride_id
    -> JOIN Feedback F ON U.Visitor_number = F.Visitor_number
    ->                 AND U.Feedback_number = F.Feedback_number
    -> GROUP BY R.Name
    -> ORDER BY Average_Rating DESC;
+-----------------+----------------+
| Ride_Name       | Average_Rating |
+-----------------+----------------+
| Thunder Coaster |         5.0000 |
+-----------------+----------------+
1 row in set (0.00 sec)
```



```
mysql>
mysql> -- 3. NESTED QUERY (Subquery)
mysql> -- (Finds all visitors who are older than the average visitor age)
mysql> SELECT
    ->     Name,
    ->     Age
    -> FROM Visitor
    -> WHERE Age > (SELECT AVG(Age) FROM Visitor);
+-------------+------+
| Name        | Age  |
+-------------+------+
| Alice Smith |   28 |
| Bob Johnson |   35 |
+-------------+------+
2 rows in set (0.00 sec)

mysql> -- Changes the delimiter from ; to $$
mysql> DELIMITER $$
mysql>
mysql> -- PROCEDURE 1: Add a new visitor and create a card for them
mysql> -- (This is based on your Functional Requirement FR2)
mysql> CREATE PROCEDURE sp_AddNewVisitor(
    ->     IN visitorNumber INT,
    ->     IN visitorName VARCHAR(150),
    ->     IN visitorAge INT,
    ->     IN cardID VARCHAR(50)
    -> )
    -> BEGIN
    ->     -- Add the visitor
    ->     INSERT INTO Visitor (Number, Name, Age)
    ->     VALUES (visitorNumber, visitorName, visitorAge);
    ->
    ->     -- Create a blank card for them with 0 balance
    ->     INSERT INTO Card (Visitor_number, Card_id, Balance, VIP)
    ->     VALUES (visitorNumber, cardID, 0.00, false);
    -> END $$
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> -- Changes the delimiter back to ;
mysql> DELIMITER ;
mysql> CALL sp_AddNewVisitor(4, 'Emily Davis', 25, 'C-004');
Query OK, 1 row affected (0.01 sec)
```

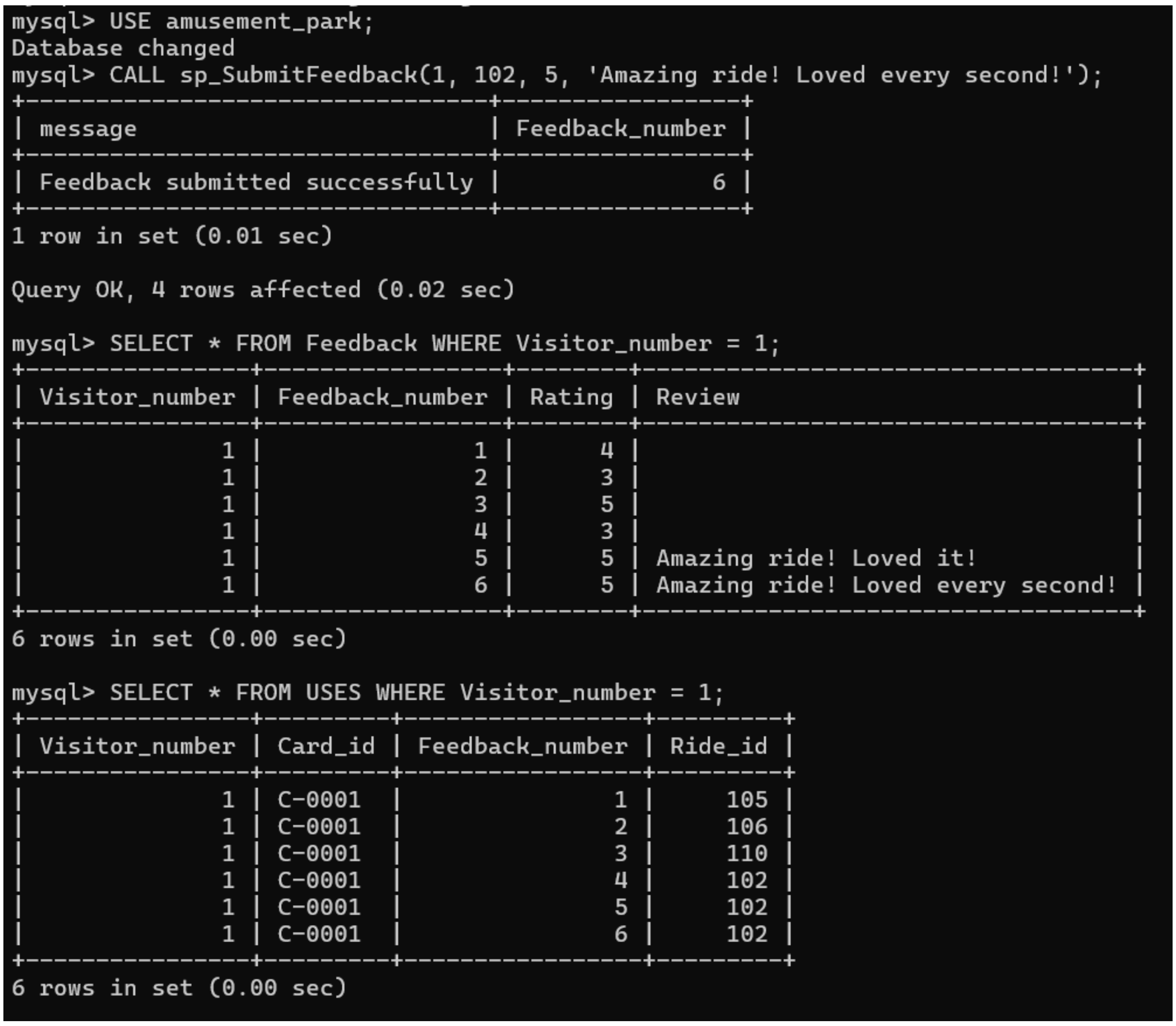


```
mysql>
mysql> -- Changes the delimiter back to ;
mysql> DELIMITER ;
mysql> CALL sp_AddNewVisitor(4, 'Emily Davis', 25, 'C-004');
Query OK, 1 row affected (0.01 sec)

mysql>
mysql> -- Run this to see if it worked!
mysql> SELECT * FROM Visitor WHERE Number = 4;
+--------+-------------+------+---------+
| Number | Name        | Age  | Eats_at |
+--------+-------------+------+---------+
|      4 | Emily Davis |   25 | NULL    |
+--------+-------------+------+---------+
1 row in set (0.00 sec)

mysql> SELECT * FROM Card WHERE Visitor_number = 4;
+----------------+---------+---------+------+
| Visitor_number | Card_id | Balance | VIP  |
+----------------+---------+---------+------+
|              4 | C-004   |    0.00 |    0 |
+----------------+---------+---------+------+
1 row in set (0.00 sec)

mysql> -- Change the delimiter
mysql> DELIMITER $$
mysql>
mysql> -- TRIGGER 1: Check VIP status before letting a visitor ride
mysql> -- (This is based on your Functional Requirement FR4)
mysql> CREATE TRIGGER trg_CheckVIPStatus
    -> BEFORE INSERT ON Enjoys
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Find the ride's VIP status
    ->     SELECT VIP INTO @is_ride_vip FROM Ride WHERE ID = NEW.Ride_id;
    ->
    ->     -- Find the visitor's VIP status from their card
    ->     SELECT VIP INTO @is_visitor_vip FROM Card WHERE Visitor_number = NEW.Visitor_number;
    ->
    ->     -- If ride is VIP (true) but visitor is not (false), block the insert
    ->     IF @is_ride_vip = true AND @is_visitor_vip = false THEN
    ->         -- This throws a custom error and stops the insert
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Visitor is not VIP. Cannot access this ride.';
    ->     END IF;
    -> END $$
Query OK, 0 rows affected (0.02 sec)
```

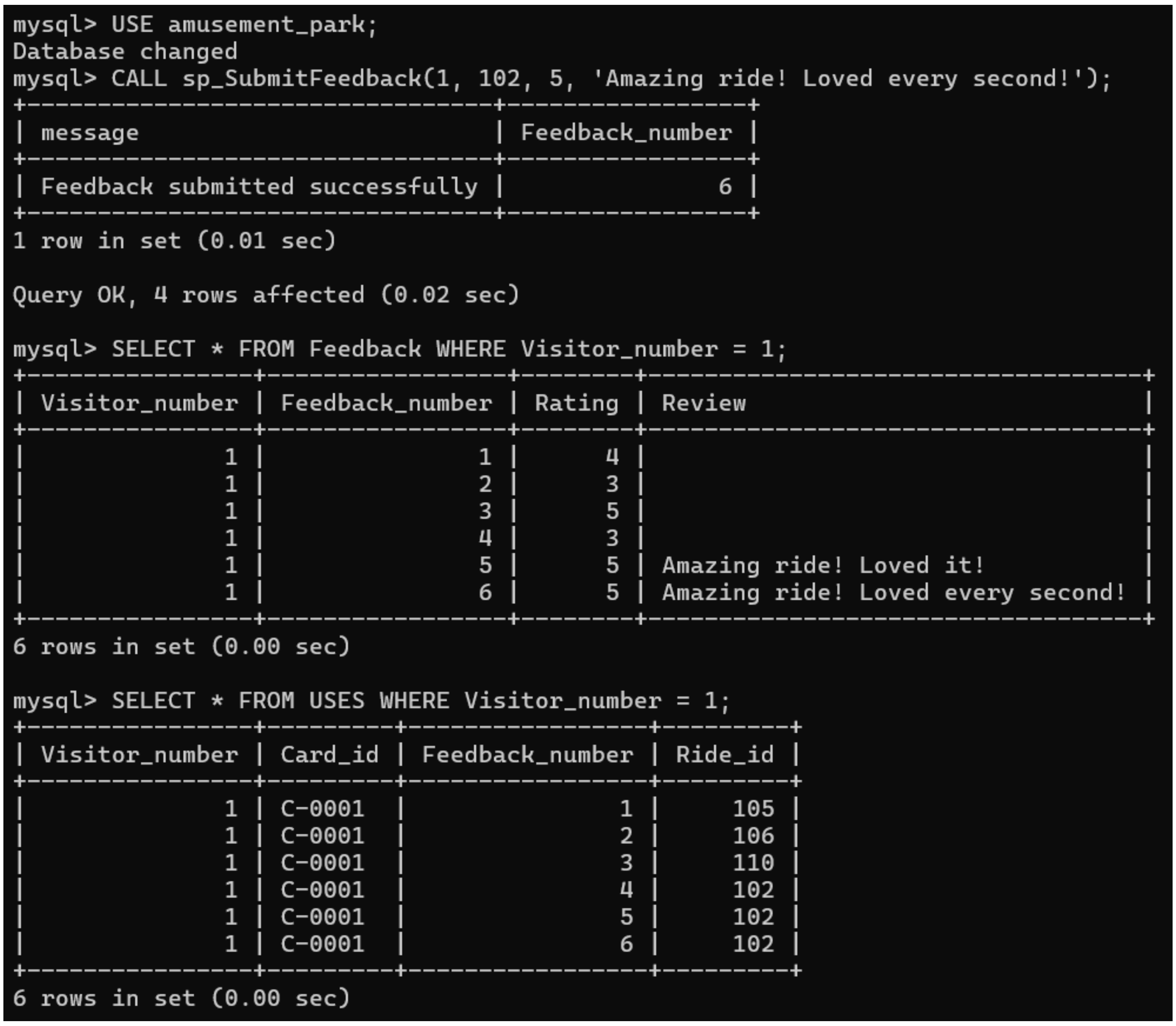


```
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> -- Change the delimiter back
mysql> DELIMITER ;
mysql> -- TEST 1: This should FAIL
mysql> -- Visitor 1 (not VIP) tries to ride Ride 103 (VIP Gold Lounge)
mysql> INSERT INTO Enjoys (Visitor_number, Ride_id) VALUES (1, 103);
ERROR 1644 (45000): Visitor is not VIP. Cannot access this ride.
mysql> -- You should get an error: "Visitor is not VIP. Cannot access this ride."
mysql>
mysql>
mysql> -- TEST 2: This should SUCCEED
mysql> -- Visitor 3 (who is VIP) tries to ride Ride 103 (VIP Gold Lounge)
mysql> INSERT INTO Enjoys (Visitor_number, Ride_id) VALUES (3, 103);
Query OK, 1 row affected (0.00 sec)

mysql> -- This should work fine.
mysql>
mysql>
mysql> -- TEST 3: This should SUCCEED
mysql> -- Visitor 1 (not VIP) tries to ride Ride 102 (Ferris Wheel, not VIP)
mysql> INSERT INTO Enjoys (Visitor_number, Ride_id) VALUES (1, 102);
ERROR 1062 (23000): Duplicate entry '1-102' for key 'enjoys.PRIMARY'
mysql> -- This should work fine.
mysql> show tables;
ERROR 2013 (HY000): Lost connection to MySQL server during query
No connection. Trying to reconnect...
Connection id:    23
Current database: amusement_park

+--------------------------+
| Tables_in_amusement_park |
+--------------------------+
| buys                     |
| card                     |
| employee                 |
| enjoys                   |
| feedback                 |
| mess                     |
| mess_food                |
| ride                     |
| shop                     |
| uses                     |
| visitor                  |
+--------------------------+
11 rows in set (0.21 sec)
```

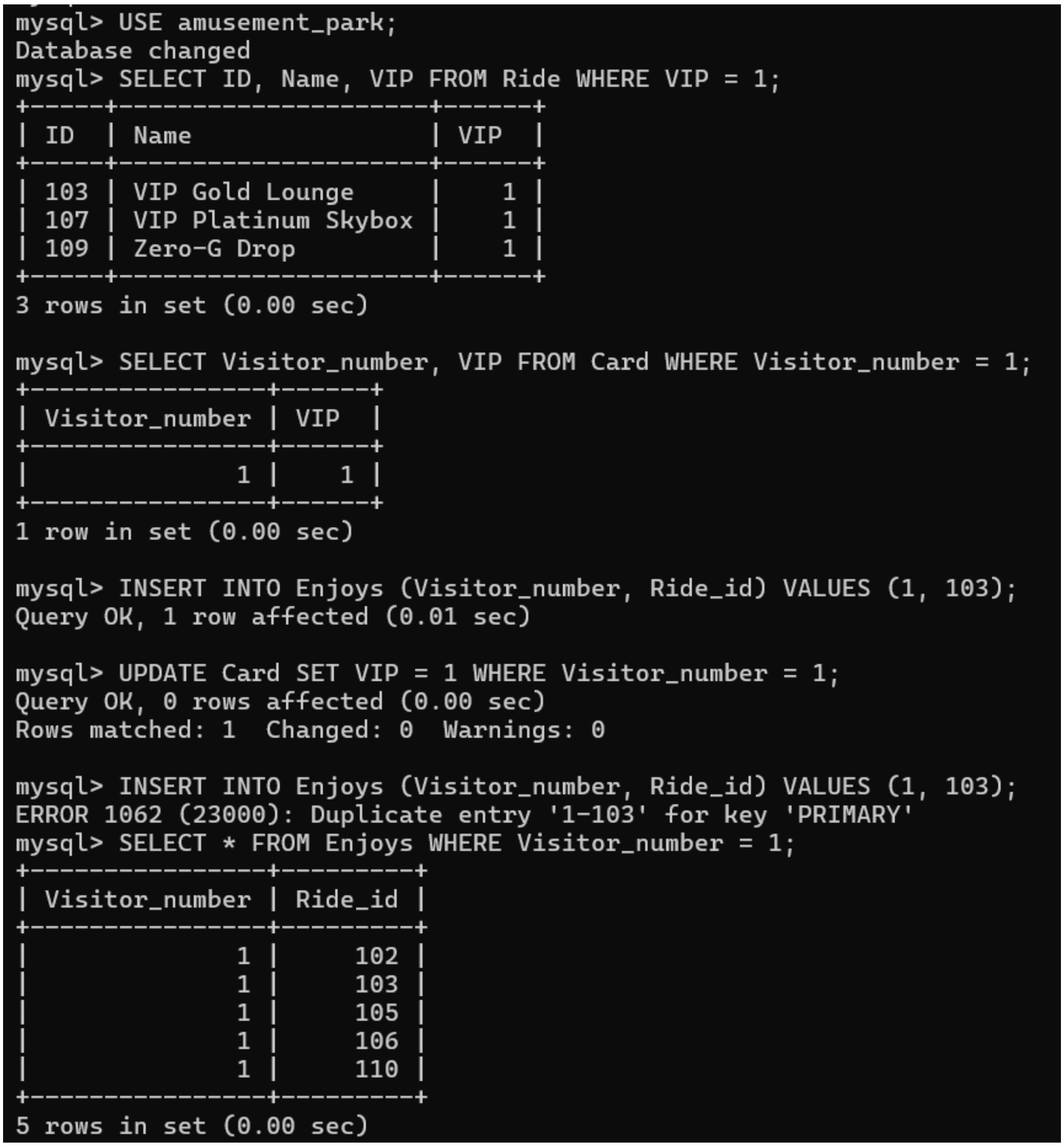
CODE SNIPPETS FOR INVOKING THE PROCEDURES/FUNCTIONS/TRIGGER



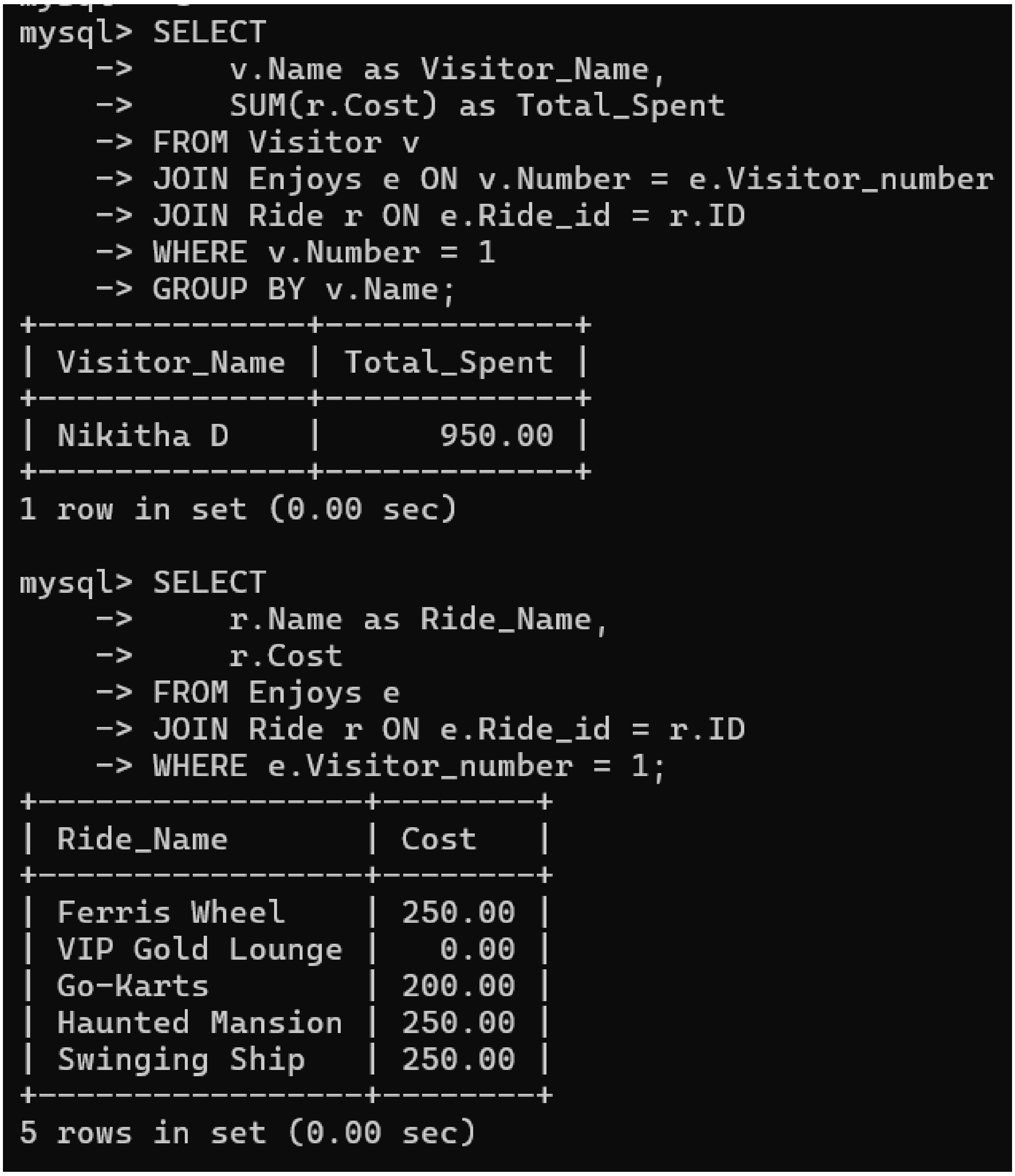
```
mysql> USE amusement_park;
Database changed
mysql> CALL sp_SubmitFeedback(1, 102, 5, 'Amazing ride! Loved every second!');
+-----------------------------------+-----------------+
| message                           | Feedback_number |
+-----------------------------------+-----------------+
| Feedback submitted successfully   |               6 |
+-----------------------------------+-----------------+
1 row in set (0.01 sec)

Query OK, 4 rows affected (0.02 sec)

mysql> SELECT * FROM Feedback WHERE Visitor_number = 1;
+----------------+-----------------+--------+-----------------------------------+
| Visitor_number | Feedback_number | Rating | Review                            |
+----------------+-----------------+--------+-----------------------------------+
|              1 |               1 |      4 |                                   |
|              1 |               2 |      3 |                                   |
|              1 |               3 |      5 |                                   |
|              1 |               4 |      3 |                                   |
|              1 |               5 |      5 | Amazing ride! Loved it!           |
|              1 |               6 |      5 | Amazing ride! Loved every second! |
+----------------+-----------------+--------+-----------------------------------+
6 rows in set (0.00 sec)

mysql> SELECT * FROM USES WHERE Visitor_number = 1;
+----------------+---------+-----------------+---------+
| Visitor_number | Card_id | Feedback_number | Ride_id |
+----------------+---------+-----------------+---------+
|              1 | C-0001  |               1 |     105 |
|              1 | C-0001  |               2 |     106 |
|              1 | C-0001  |               3 |     110 |
|              1 | C-0001  |               4 |     102 |
|              1 | C-0001  |               5 |     102 |
|              1 | C-0001  |               6 |     102 |
+----------------+---------+-----------------+---------+
6 rows in set (0.00 sec)
```



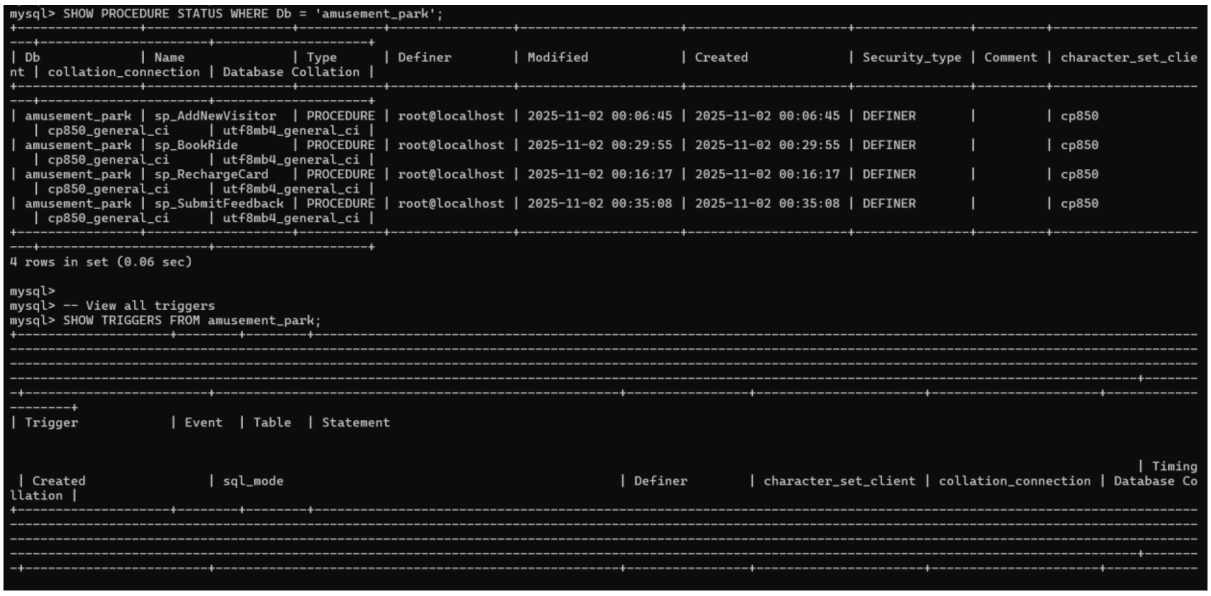
```
mysql> USE amusement_park;
Database changed
mysql> CALL sp_SubmitFeedback(1, 102, 5, 'Amazing ride! Loved every second!');
+--------------------------------+-----------------+
| message                        | Feedback_number |
+--------------------------------+-----------------+
| Feedback submitted successfully |               6 |
+--------------------------------+-----------------+
1 row in set (0.01 sec)

Query OK, 4 rows affected (0.02 sec)

mysql> SELECT * FROM Feedback WHERE Visitor_number = 1;
+----------------+-----------------+--------+-----------------------------------+
| Visitor_number | Feedback_number | Rating | Review                            |
+----------------+-----------------+--------+-----------------------------------+
|              1 |               1 |      4 |                                   |
|              1 |               2 |      3 |                                   |
|              1 |               3 |      5 |                                   |
|              1 |               4 |      3 |                                   |
|              1 |               5 |      5 | Amazing ride! Loved it!           |
|              1 |               6 |      5 | Amazing ride! Loved every second! |
+----------------+-----------------+--------+-----------------------------------+
6 rows in set (0.00 sec)

mysql> SELECT * FROM USES WHERE Visitor_number = 1;
+----------------+---------+-----------------+---------+
| Visitor_number | Card_id | Feedback_number | Ride_id |
+----------------+---------+-----------------+---------+
|              1 | C-0001  |               1 |     105 |
|              1 | C-0001  |               2 |     106 |
|              1 | C-0001  |               3 |     110 |
|              1 | C-0001  |               4 |     102 |
|              1 | C-0001  |               5 |     102 |
|              1 | C-0001  |               6 |     102 |
+----------------+---------+-----------------+---------+
6 rows in set (0.00 sec)
```



```
mysql> USE amusement_park;
Database changed
mysql> SELECT ID, Name, VIP FROM Ride WHERE VIP = 1;
+-----+---------------------+------+
| ID  | Name                | VIP  |
+-----+---------------------+------+
| 103 | VIP Gold Lounge     |    1 |
| 107 | VIP Platinum Skybox |    1 |
| 109 | Zero-G Drop         |    1 |
+-----+---------------------+------+
3 rows in set (0.00 sec)

mysql> SELECT Visitor_number, VIP FROM Card WHERE Visitor_number = 1;
+----------------+------+
| Visitor_number | VIP  |
+----------------+------+
|              1 |    1 |
+----------------+------+
1 row in set (0.00 sec)

mysql> INSERT INTO Enjoys (Visitor_number, Ride_id) VALUES (1, 103);
Query OK, 1 row affected (0.01 sec)

mysql> UPDATE Card SET VIP = 1 WHERE Visitor_number = 1;
Query OK, 0 rows affected (0.00 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> INSERT INTO Enjoys (Visitor_number, Ride_id) VALUES (1, 103);
ERROR 1062 (23000): Duplicate entry '1-103' for key 'PRIMARY'
mysql> SELECT * FROM Enjoys WHERE Visitor_number = 1;
+----------------+---------+
| Visitor_number | Ride_id |
+----------------+---------+
|              1 |     102 |
|              1 |     103 |
|              1 |     105 |
|              1 |     106 |
|              1 |     110 |
+----------------+---------+
5 rows in set (0.00 sec)
```



```
mysql> SELECT
    ->     v.Name as Visitor_Name,
    ->     SUM(r.Cost) as Total_Spent
    -> FROM Visitor v
    -> JOIN Enjoys e ON v.Number = e.Visitor_number
    -> JOIN Ride r ON e.Ride_id = r.ID
    -> WHERE v.Number = 1
    -> GROUP BY v.Name;
+--------------+-------------+
| Visitor_Name | Total_Spent |
+--------------+-------------+
| Nikitha D    |      950.00 |
+--------------+-------------+
1 row in set (0.00 sec)

mysql> SELECT
    ->     r.Name as Ride_Name,
    ->     r.Cost
    -> FROM Enjoys e
    -> JOIN Ride r ON e.Ride_id = r.ID
    -> WHERE e.Visitor_number = 1;
+-----------------+--------+
| Ride_Name       | Cost   |
+-----------------+--------+
| Ferris Wheel    | 250.00 |
| VIP Gold Lounge |   0.00 |
| Go-Karts        | 200.00 |
| Haunted Mansion | 250.00 |
| Swinging Ship   | 250.00 |
+-----------------+--------+
5 rows in set (0.00 sec)
```



```
mysql> SHOW PROCEDURE STATUS WHERE Db = 'amusement_park';
+----------------+-------------------+-----------+----------------+---------------------+---------------------+---------------+---------+-------------------
---+----------------------+--------------------+
| Db             | Name              | Type      | Definer        | Modified            | Created             | Security_type | Comment | character_set_clie
nt | collation_connection | Database Collation |
+----------------+-------------------+-----------+----------------+---------------------+---------------------+---------------+---------+-------------------
---+----------------------+--------------------+
| amusement_park | sp_AddNewVisitor  | PROCEDURE | root@localhost | 2025-11-02 00:06:45 | 2025-11-02 00:06:45 | DEFINER       |         | cp850
   | cp850_general_ci     | utf8mb4_general_ci |
| amusement_park | sp_BookRide       | PROCEDURE | root@localhost | 2025-11-02 00:29:55 | 2025-11-02 00:29:55 | DEFINER       |         | cp850
   | cp850_general_ci     | utf8mb4_general_ci |
| amusement_park | sp_RechargeCard   | PROCEDURE | root@localhost | 2025-11-02 00:16:17 | 2025-11-02 00:16:17 | DEFINER       |         | cp850
   | cp850_general_ci     | utf8mb4_general_ci |
| amusement_park | sp_SubmitFeedback | PROCEDURE | root@localhost | 2025-11-02 00:35:08 | 2025-11-02 00:35:08 | DEFINER       |         | cp850
   | cp850_general_ci     | utf8mb4_general_ci |
+----------------+-------------------+-----------+----------------+---------------------+---------------------+---------------+---------+-------------------
---+----------------------+--------------------+
4 rows in set (0.06 sec)

mysql>
mysql> -- View all triggers
mysql> SHOW TRIGGERS FROM amusement_park;
+---------------------+--------+--------+-----------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------+-------
-+------------------------+-----------------------------------------------------+----------------+----------------------+----------------------+----------
--------+
| Trigger             | Event  | Table  | Statement


                                                                                                                                                    | Timing
 | Created                | sql_mode                                            | Definer        | character_set_client | collation_connection | Database Co
llation |
+---------------------+--------+--------+-----------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------+-------
-+------------------------+-----------------------------------------------------+----------------+----------------------+----------------------+----------
```

GITHUB REPOSITORY LINK:

https://github.com/PES2UG23CS388/DBMS_Mini_Project_Amusement_park_366_388