Albert DeLaCruz

Nino Arias

Joshua Padilla

Alfonso Arguello Ramirez

ISM 4210 Group 7 Project: SmartPark

“The better parking solution”

Table of Contents

Introduction

The business in focus is a modern, technology-driven parking garage system named "SmartPark Solutions." SmartPark Solutions aims to provide safe, efficient, and convenient parking solutions for urban areas experiencing high vehicle density. The parking garage will leverage automated systems, real-time data monitoring, and mobile applications to optimize parking availability and enhance user experience. Located in a bustling downtown area, SmartPark Solutions targets daily commuters, local businesses, and event attendees who struggle with finding reliable parking options.

Key features of SmartPark Solutions include:

- Automated ticketing and transaction systems.
- Real-time parking availability updates via a mobile app.
- Advanced security measures, including surveillance and license plate recognition.
- Compact, vertical parking designs to maximize space utilization.

By combining cutting-edge technology and customer-centric services, SmartPark Solutions aims to redefine urban parking experiences and alleviate common pain points associated with traditional parking facilities.

Underlying Business Problem

Urban areas are increasingly plagued by a lack of adequate parking spaces, leading to several challenges:

1. **Congestion and Wasted Time:** Drivers spend significant amounts of time searching for parking, which contributes to traffic congestion and increases commute times.
2. **Environmental Impact:** Prolonged idling and unnecessary driving in search of parking contribute to increased carbon emissions.
3. **Customer Frustration:** The unavailability of convenient parking spaces often frustrates drivers, potentially deterring them from visiting certain areas or businesses.
4. **Inefficient Space Usage:** Many traditional parking garages are poorly designed, leading to underutilization of available space.

Reasons for the Selection of the Business:

The decision to establish SmartPark Solutions stems from several compelling factors:

1. **High Demand in Urban Areas:** The increasing population density and vehicle ownership in cities have created a consistent demand for innovative parking solutions.
2. **Market Opportunity:** Current parking solutions are outdated and lack technological integration, presenting a gap in the market that SmartPark Solutions can fill.
3. **Revenue Potential:** A well-managed parking garage in a prime location offers a steady stream of revenue through daily fees, subscriptions, and premium services.
4. **Enhanced Customer Experience:** By addressing common frustrations with traditional parking, SmartPark Solutions can build a loyal customer base and establish itself as a trusted brand.

SmartPark Solutions is uniquely positioned to address these challenges and capitalize on the opportunities within the urban parking sector, providing value to both customers and the community.

Database Design

Business Rules

Parking Spots

1. Each parking spot must have a unique identifier (`SpotID`).
2. A parking spot can either be **occupied** or **unoccupied** (`IsOccupied`).
3. A parking spot can be associated with multiple parking transactions over time, but only one active transaction at a time.

Vehicles

4. Each vehicle must have a unique identifier (`VehicleID`).
5. A vehicle can only occupy one parking spot at a time.
6. A vehicle must be linked to a parking transaction to use the parking system.
7. Each vehicle may have multiple subscriptions, but only one active subscription for a given time period.

ParkingTransactions

8. Each parking transaction must have a unique identifier (`TransactionID`).
9. Every parking transaction must be linked to a specific vehicle (`VehicleID`) and a specific parking spot (`SpotID`).
10. The parking transaction must record the **check-in time**, and the **check-out time** is optional until the vehicle leaves the spot.
11. A parking transaction must include the total amount paid (`AmountPaid`), calculated based on the duration of parking and any applicable rates.

Subscriptions

12. Each subscription must have a unique identifier (`SubscriptionID`).
13. A subscription is optional but, if created, must be linked to a specific vehicle (`VehicleID`).
14. Subscriptions must specify a **start date** (`StartDate`), an **end date** (`EndDate`), and the **type** of subscription (`SubscriptionType`, e.g., monthly or yearly).
15. A vehicle with an active subscription is exempt from individual parking transaction payments but still records transactions for tracking purposes.

Entities:

ParkingSpots

- **Attributes:**
  - `SpotID (PK)`: Unique identifier for each parking spot.
  - `SpotNumber`: The physical number assigned to a parking spot for easy identification.
  - `IsOccupied`: A boolean value (e.g., `True` or `False`) indicating whether the spot is currently occupied.
- **Purpose:** Tracks each parking spot in the system and its availability status.

Vehicles

- **Attributes:**
  - `VehicleID (PK)`: Unique identifier for each registered vehicle in the system.

    - `LicensePlate`: The license plate number of the vehicle.
    - `VehicleType`: The type of vehicle (e.g., sedan, SUV, EV).
    - `OwnerName`: Name of the vehicle owner.
    - `ContactNumber`: The contact number of the vehicle owner.
- **Purpose:** Maintains information about vehicles using the parking facility and their owners.

ParkingTransactions

- **Attributes:**
    - `TransactionID (PK)`: Unique identifier for each parking transaction.
    - `VehicleID (FK)`: Foreign key linking to the `Vehicles` entity.
    - `SpotID (FK)`: Foreign key linking to the `ParkingSpots` entity.
    - `CheckInTime`: The timestamp when a vehicle checks into a parking spot.
    - `CheckOutTime`: The timestamp when a vehicle checks out of the parking spot.
    - `AmountPaid`: The total amount paid for the parking transaction.
- **Purpose:** Records every instance of a vehicle parking in the system, including the parking spot used, the vehicle involved, and payment details.

Subscriptions

- **Attributes:**
    - `SubscriptionID (PK)`: Unique identifier for each subscription.
    - `VehicleID (FK)`: Foreign key linking to the `Vehicles` entity.
    - `StartDate`: The starting date of the subscription.
    - `EndDate`: The ending date of the subscription.

    - `SubscriptionType`: The type of subscription (e.g., monthly, yearly).
    - `AmountPaid`: The amount paid for the subscription.
- **Purpose:** Tracks subscription plans for vehicles, providing recurring access to the parking system.

Relationships:

ParkingSpots to ParkingTransactions

- **Type:** 1-to-Many
    - **Explanation:** A parking spot can be associated with multiple parking transactions over time, but each transaction involves only one parking spot.
    - **Key Mapping:** `SpotID` in `ParkingTransactions` is a foreign key referencing `SpotID` in `ParkingSpots`.

Vehicles to ParkingTransactions

- **Type:** 1-to-Many
    - **Explanation:** A vehicle can have multiple parking transactions over time, but each parking transaction is associated with one vehicle.
    - **Key Mapping:** `VehicleID` in `ParkingTransactions` is a foreign key referencing `VehicleID` in `Vehicles`.

Vehicles to Subscriptions

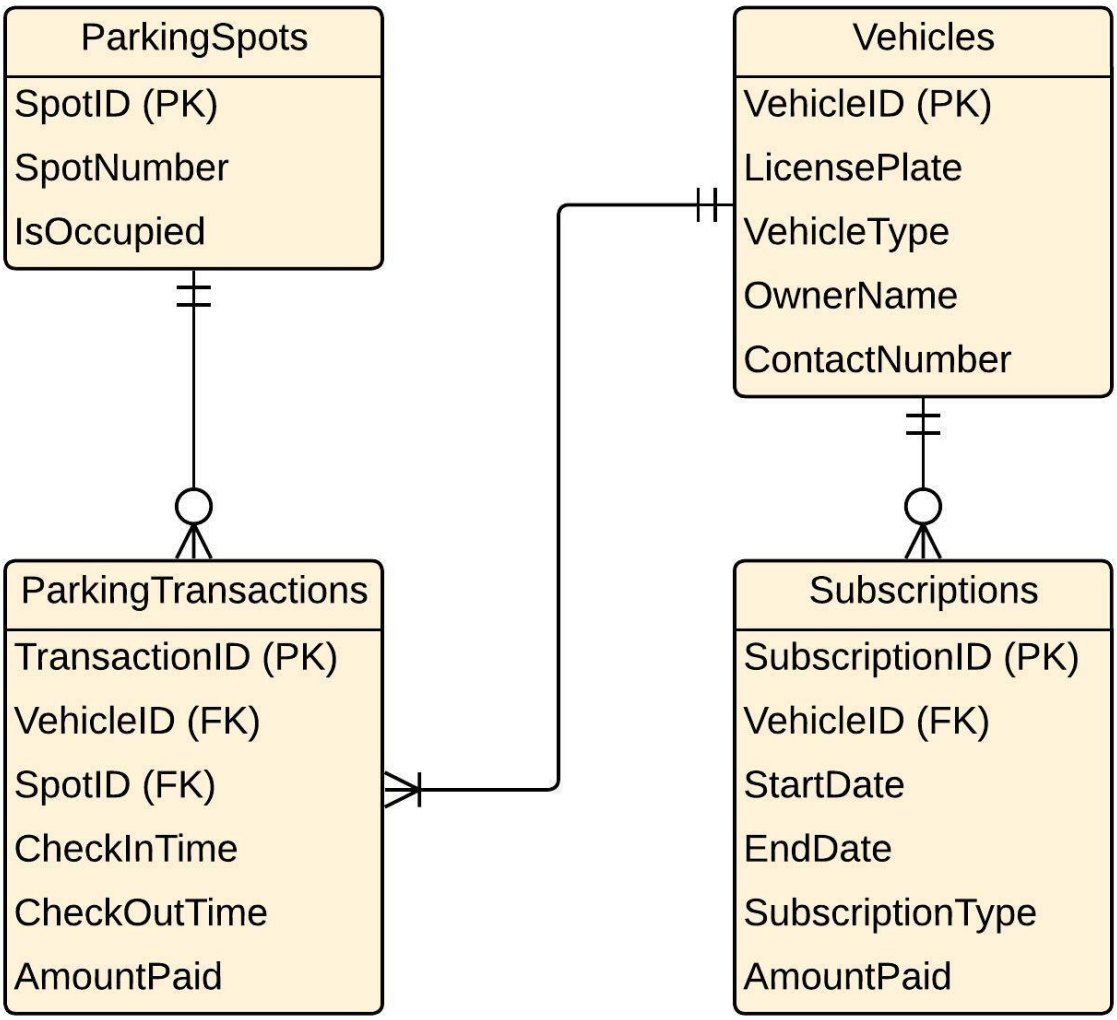
- **Type:** 1-to-Many
    - **Explanation:** A vehicle can have multiple subscriptions (e.g., a new subscription after the previous one ends), but each subscription is linked to one specific vehicle.

- **Key Mapping:** `VehicleID` in `Subscriptions` is a foreign key referencing `VehicleID` in `Vehicles`.

This database structure is designed to efficiently manage and reflect open or taken parking spaces, track transactions, and handle different kinds of subscriptions while maintaining data integrity through relationships between entities.

Entity Relationship Diagram

SMARTPARK



Database Implementation

SQL CODE

```sql
CREATE DATABASE ParkingGarageDB;
USE ParkingGarageDB;

-- Table to store information about parking spots
CREATE TABLE ParkingSpots (
    SpotID INT AUTO_INCREMENT PRIMARY KEY,
    SpotNumber VARCHAR(10) NOT NULL,
    Level INT NOT NULL,
    IsOccupied BOOLEAN NOT NULL DEFAULT FALSE
);

-- Table to store information about vehicles
CREATE TABLE Vehicles (
    VehicleID INT AUTO_INCREMENT PRIMARY KEY,
    LicensePlate VARCHAR(15) NOT NULL UNIQUE,
    VehicleType ENUM('Car', 'Truck', 'Motorcycle') NOT NULL,
    OwnerName VARCHAR(100),
    ContactNumber VARCHAR(15)
);

-- Table to store parking transactions
CREATE TABLE ParkingTransactions (
    TransactionID INT AUTO_INCREMENT PRIMARY KEY,
    VehicleID INT NOT NULL,
    SpotID INT NOT NULL,
    CheckInTime DATETIME NOT NULL,
    CheckOutTime DATETIME,
    AmountPaid DECIMAL(10, 2),
    FOREIGN KEY (VehicleID) REFERENCES Vehicles(VehicleID),
    FOREIGN KEY (SpotID) REFERENCES ParkingSpots(SpotID)
);

```

```sql
-- Table to manage subscriptions for regular customers
CREATE TABLE Subscriptions (
    SubscriptionID INT AUTO_INCREMENT PRIMARY KEY,
    VehicleID INT NOT NULL,
    StartDate DATE NOT NULL,
    EndDate DATE NOT NULL,
    SubscriptionType ENUM('Monthly', 'Yearly') NOT NULL,
    AmountPaid DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (VehicleID) REFERENCES Vehicles(VehicleID)
);

-- Insert some sample data
INSERT INTO ParkingSpots (SpotNumber, Level, IsOccupied) VALUES
('A1', 1, FALSE),
('A2', 1, FALSE),
('B1', 2, FALSE),
('B2', 2, FALSE),
('C1', 3, FALSE),
('C2', 3, FALSE),
('D1', 4, FALSE),
('D2', 4, FALSE);

INSERT INTO Vehicles (LicensePlate, VehicleType, OwnerName, ContactNumber) VALUES
('ABC123', 'Car', 'John Doe', '123-456-7890'),
('XYZ789', 'Motorcycle', 'Jane Smith', '098-765-4321'),
('LMN456', 'Truck', 'Alice Johnson', '456-789-0123'),
('PQR321', 'Car', 'Bob Brown', '789-012-3456');

INSERT INTO ParkingTransactions (VehicleID, SpotID, CheckInTime, CheckOutTime, AmountPaid) VALUES
(1, 1, '2024-12-01 08:00:00', '2024-12-01 12:00:00', 20.00),
(2, 2, '2024-12-01 09:00:00', NULL, NULL),
(3, 3, '2024-12-01 10:00:00', '2024-12-01 14:00:00', 25.00),
```

Table Creation

Table: parkingspots

Columns:

SpotID	Primary Key
SpotNumber	varchar(10)
Level	int
IsOccupied	tinyint(1)

Table: parking transactions

Columns:

TransactionID	(Primary Key)
VehicleID	int
SpotID	int
CheckInTime	datetime
CheckOutTime	datetime
AmountPaid	decimal(10,2)

Table: subscriptions

Columns:

SubscriptionID	(Primary Key)
VehicleID	int
StartDate	date
EndDate	date
SubscriptionType	enum('Monthly','Yearly')
AmountPaid	decimal(10,2)

Table:vehicles

Columns:

VehicleID	(Primary Key)
LicensePlate	varchar(15)
VehicleType	enum('Car','Truck','Motorcycle')
OwnerName	varchar(100)
ContactNumber	varchar(15)

SQL Query

The objective of the queries being provided below are to give it a hypothetical situation in which we can see from Smart Park deal with in certain situations using the relational database in MySQL. Provided below will be 10 objectives used:

1.List of all parking spots and their current status

```
SELECT SpotNumber, Level, IsOccupied
FROM ParkingSpots;
```

2. Retrieve vehicles with yearly subscriptions

```
SELECT v.LicensePlate, v.OwnerName, s.SubscriptionType, s.StartDate, s.EndDate
FROM Subscriptions s
JOIN Vehicles v ON s.VehicleID = v.VehicleID
WHERE s.SubscriptionType = 'Yearly';
```

3. Find the total number of vehicles currently parked

```
SELECT COUNT(*) AS CurrentlyParkedVehicles
FROM ParkingTransactions
WHERE CheckOutTime IS NULL;
```

4. Get a list of transactions for the current day

```
SELECT pt.TransactionID, v.LicensePlate, ps.SpotNumber, pt.CheckInTime, pt.CheckOutTime,
pt.AmountPaid
FROM ParkingTransactions pt
JOIN Vehicles v ON pt.VehicleID = v.VehicleID
JOIN ParkingSpots ps ON pt.SpotID = ps.SpotID
WHERE DATE(pt.CheckInTime) = CURDATE();
```

5. Retrieve all vehicles parked on Level 2

```
SELECT pt.TransactionID, v.LicensePlate, ps.SpotNumber, pt.CheckInTime, pt.CheckOutTime,
pt.AmountPaid
FROM ParkingTransactions pt
JOIN Vehicles v ON pt.VehicleID = v.VehicleID
JOIN ParkingSpots ps ON pt.SpotID = ps.SpotID
WHERE DATE(pt.CheckInTime) = CURDATE();
```

6. Calculate total revenue from monthly subscriptions

```
SELECT SUM(s.AmountPaid) AS TotalMonthlySubscriptionRevenue
FROM Subscriptions s
WHERE s.SubscriptionType = 'Monthly';
```

7. Identify vehicles that parked for more than 4 hours in a single transaction

```
SELECT v.LicensePlate, pt.CheckInTime, pt.CheckOutTime, TIMESTAMPDIFF(HOUR,
pt.CheckInTime, pt.CheckOutTime) AS ParkingDuration
FROM ParkingTransactions pt
JOIN Vehicles v ON pt.VehicleID = v.VehicleID
WHERE pt.CheckOutTime IS NOT NULL
AND TIMESTAMPDIFF(HOUR, pt.CheckInTime, pt.CheckOutTime) > 4;
```

8. Find parking spots that are currently vacant

```
SELECT SpotNumber, Level
FROM ParkingSpots
WHERE IsOccupied = FALSE;
```

9. List all vehicles that have not checked out yet

```
SELECT v.LicensePlate, v.OwnerName, pt.CheckInTime, ps.SpotNumber, ps.Level
FROM ParkingTransactions pt
JOIN Vehicles v ON pt.VehicleID = v.VehicleID
JOIN ParkingSpots ps ON pt.SpotID = ps.SpotID
WHERE pt.CheckOutTime IS NULL;
```

10. Get a summary of revenue generated by vehicle type

```
SELECT v.VehicleType, SUM(pt.AmountPaid) AS TotalRevenue
FROM ParkingTransactions pt
JOIN Vehicles v ON pt.VehicleID = v.VehicleID
WHERE pt.AmountPaid IS NOT NULL
GROUP BY v.VehicleType;
```

Conclusion

The SmartPark Solutions project offers a comprehensive and innovative approach to addressing the challenges associated with urban parking. By

leveraging advanced database management systems and integrating technology-driven features, the project highlights the ability to:

1. Optimize parking availability through real-time monitoring and mobile app integrations.
2. Reduce environmental impact by minimizing idling and unnecessary driving.
3. Enhance the customer experience with automated systems, subscription models, and advanced security measures.

The database design and implementation demonstrate a structure capable of efficiently managing parking transactions, vehicle data, and subscription plans. SQL queries provide actionable insights, such as identifying parking spot availability, monitoring revenue streams, and analyzing parking behaviors, enabling SmartPark Solutions to make data-driven decisions and maintain operational excellence.

This project underscores the potential of combining modern technology with user-centric services to alleviate common urban parking issues, contributing positively to urban infrastructure and customer satisfaction.

Appendix

SQL Queries Used

1. **List of all parking spots and their current status**

```
SELECT SpotNumber, Level, IsOccupied FROM ParkingSpots;
```

2. **Retrieve vehicles with yearly subscriptions**

```
SELECT v.LicensePlate, v.OwnerName, s.SubscriptionType, s.StartDate, s.EndDate
FROM Subscriptions s
JOIN Vehicles v ON s.VehicleID = v.VehicleID
WHERE s.SubscriptionType = 'Yearly';
```

3. **Find the total number of vehicles currently parked**

```
SELECT COUNT(*) AS CurrentlyParkedVehicles
FROM ParkingTransactions
WHERE CheckOutTime IS NULL;
```

Entity-Relationship Diagram

Details of the ERD were described relationships between entities such as ParkingSpots, Vehicles, ParkingTransactions, and Subscriptions.

Glossary

- **IsOccupied**: A boolean indicator of parking spot availability.
- **VehicleID**: Unique identifier for each registered vehicle.
- **TransactionID**: Identifier for individual parking transactions.

Additional Notes

- All database tables were designed following normalization principles to reduce redundancy and ensure data integrity.
- Future iterations of the project could incorporate machine learning algorithms for predictive analytics to further enhance operational efficiency.