EZ Rental Inc.

TABLE OF CONTENT

PROJECT 1 – EZ Rental Auto Rental POS Management System Database Design and Implementation

1.Executive Summary:

The **EZ Auto Rental POS Management System** is a robust, database-driven platform developed to enhance and streamline the operations of car rental businesses, fleet managers, and automotive dealerships offering rental services. This project focuses on the **design and implementation of a fully normalized relational database** using **Microsoft SQL Server**, ensuring data integrity, optimized performance, and scalability for future growth.

Purpose and Scope

This system addresses critical rental business needs such as fleet utilization, customer data management, billing accuracy, and operational efficiency. The primary goal is to provide a backend data infrastructure that supports a comprehensive rental management system, with clearly defined relationships, constraints, and validation logic embedded into the database design.

Key Features and Capabilities

The EZ Auto Rental POS Management System offers a range of features designed to enhance operational efficiency and data reliability. It includes a comprehensive reservation management component that enables efficient handling of vehicle availability, customer bookings, and rental scheduling. The system also supports detailed vehicle and fleet tracking, with structured data storage that allows categorization by network company, issuing bank, and merchant service provider. Customer and merchant profiles are maintained through dedicated tables that store essential information about customers, issuing banks, network companies, and corporate merchant banks, allowing for detailed and relational data tracking. An automated billing framework is integrated into the system to support accurate rental charges, late fees, and monitoring of credit card limits and usage. To ensure data quality and consistency, the system applies validation rules and constraints such as CHECK, UNIQUE, NOT NULL, and PRIMARY KEY. Furthermore, the database structure is optimized to support robust reporting and querying, making it suitable for integration with analytics and business intelligence tools.

Target Users

The EZ Auto Rental POS Management System is tailored for rental agencies, dealership rental departments, and fleet managers who require a well-organized, reliable, and

scalable back-end solution. This system serves as the foundational infrastructure for a point-of-sale or web-based rental management application, offering a structured environment that supports business operations and growth.

Technical Implementation

The system is built on Microsoft SQL Server, utilizing a design approach that emphasizes entity-relationship modeling, third normal form (3NF) normalization, and domain-driven naming conventions to promote clarity and maintainability. Tables created include key entities such as CREDITCARD, CREDITCARDISSUINGBANK, CREDITCARDNETWORKCOMPANY, CREDITCARDCORPORATEMERCHANTBANK, COUNTRY, USSTATE, and others. To ensure data integrity and enforce business rules, constraints such as PRIMARY KEY, FOREIGN KEY, CHECK, NOT NULL, and UNIQUE were applied throughout the schema. The overall development strategy centered on modular table creation, consistent data enforcement, and a flexible architecture designed to accommodate future expansion and integration

Future Roadmap

The EZ Auto Rental POS Management System is designed with future enhancements that will extend its capabilities and improve overall functionality. One of the primary goals is to integrate the system with front-end applications, including point-of-sale and mobile platforms, to offer a more user-friendly and accessible interface. Automation will be further supported through the development of stored procedures and database triggers, enabling smoother operations and reducing manual intervention. Security features will be enhanced by implementing user role management and audit logging, ensuring that data access is properly controlled and monitored. To support scalability and flexibility, functionalities for importing and exporting data will be incorporated, allowing for easier data migration and system interoperability. Additionally, integration with payment gateways and external APIs will be pursued to enable seamless and secure financial transactions, rounding out the system's readiness for enterprise-level deployment and future growth.

2. Problem Statement & Objectives

The **EZRental Auto Rental Management System** features are designed to:

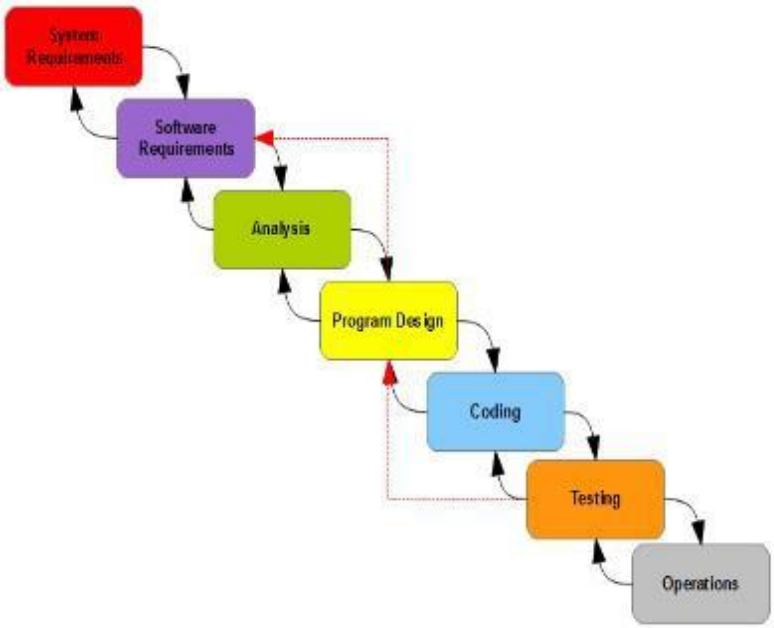
Allow customers, both retail and corporate customers, to reserve vehicles for renting, like other in-person or online car rental systems such as **Avis, Hertz, Budget,** etc.
The application needs to provide the required functionalities for our Customer Service representatives and other front-line workers in our rental agencies to service in-person customers for renting and reservation processes.
Provide the features for our business users in our corporate offices who need to create reports, perform analytics and other business functionalities related to the management of the reservations and rental of our vehicles, via our INTRENET PORTAL.
Finally, features to allow customers to make & manage vehicle reservations, profile, account etc., via the public internet.

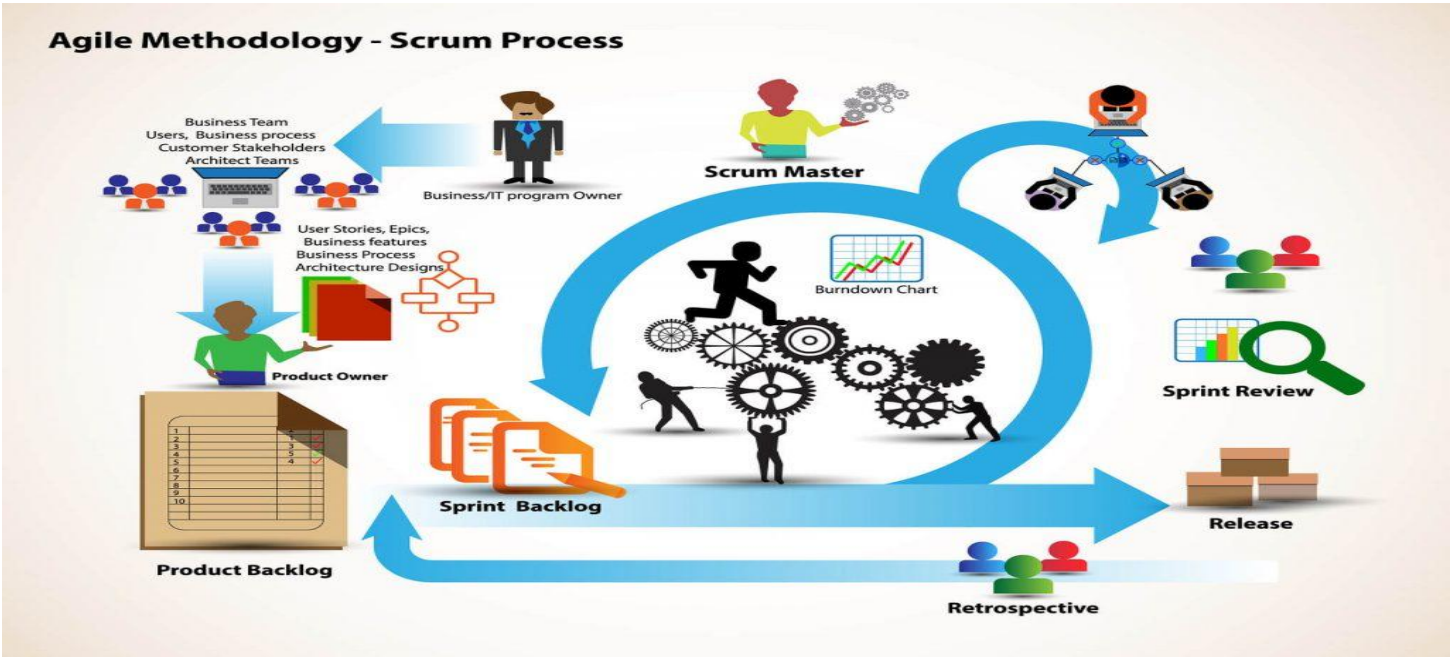
The application is designed to support dozens of major cities around the world. In addition, provide a great user experience both in the rental agencies as well as the online systems with the best competitive pricing available on the market.
The application is designed to support different branches of the company in US, Canada, Mexico, United Kingdom, Japan & Australia and kept option for expanding it in future.

3. Project Management Methodology

- For this **Auto Rental Management System project**, a high-level overview of the following two most popular Project Management Methodologies used in industry:

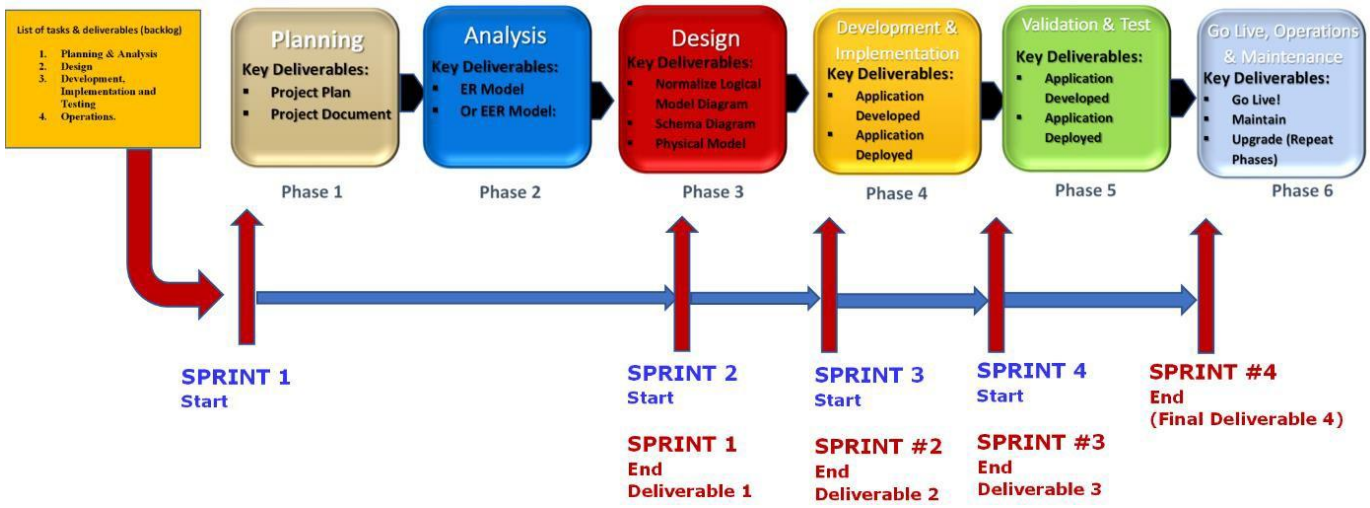
  - **Waterfall Project Management Methodology** .
  - **Agile Project Management Methodology** .





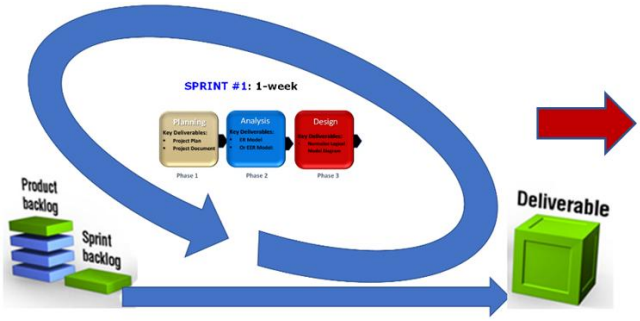
In this project we used a Combination of Waterfall & Agile Methodologies

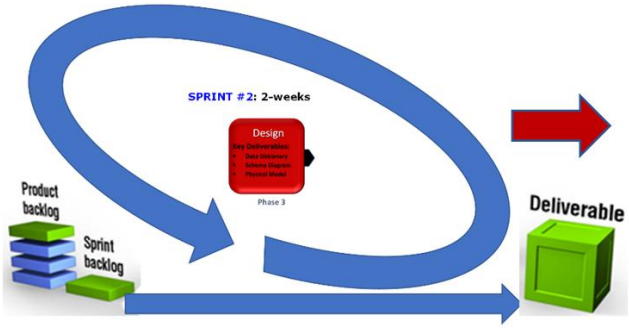
- **Here we divided the Waterfall Methodology Phases into SPRINTS for 4 SPRINTS for a period of 5-Weeks. Diagram and SPRINTS descriptions table shown below:**

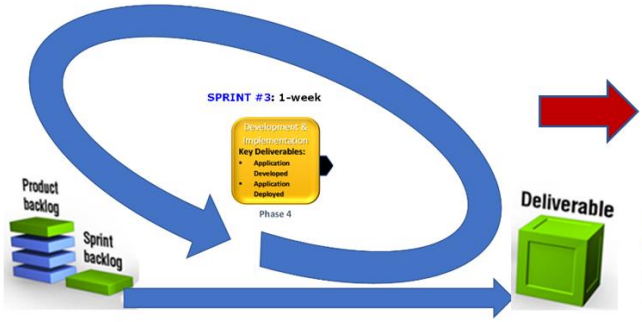


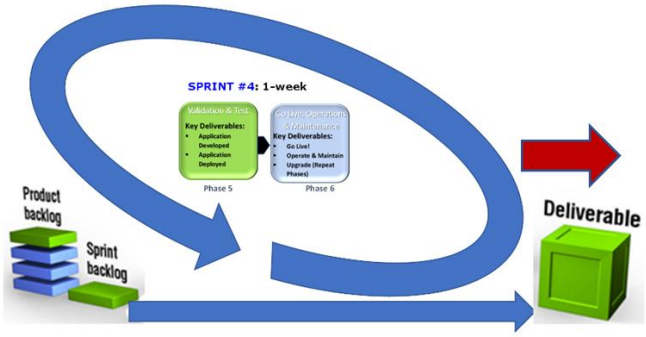
- **The table below describes each phase within an Agile SPRINT in detail:**

AGILE SPRINT #	WATERFALL PHASE	Output Deliverable
SPRINT #1	Planning	**1. Create Project Document – Formatted and populated as per requirements.** **2. Business Requirements (Included in Project Document – List of Business & Technical Requirements from customer.**
	Analysis	**3. ER/EER Conceptual Model Diagram**
	Design Phase (Part 1)	**4. Normalized Logical Model Diagram**
SPRINT #2	Design Phase (Part 2)	**5. Data Dictionary matrix** **6. Physical Schema Design Diagram – from Normalize Logical Model + DataDictionary combination.**
SPRINT #3	Development & Implementation	**7. Database application developed & implemented – This includes the Database application installed, setup and configured** **8. Generate the actual Physical Schema Diagram – from the Database & compared to the Physical Schema Design Diagram – to validate the design.**
SPRINT #4	Validation & Testing	**9. Unit & Integration testing.**
	Operations	**10. Operations – or keep database running. Keeping the lights on!**



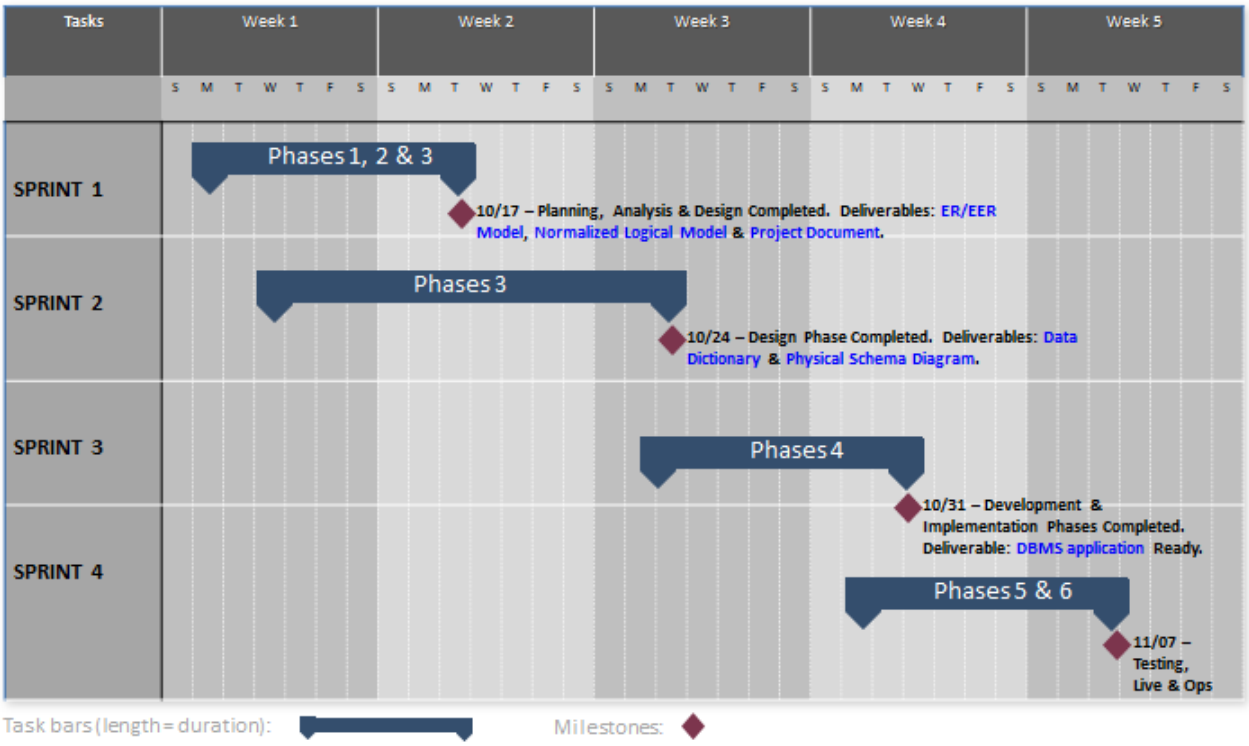






END of Project

Project Sprints, Phases, Timelines & Milestones



4. Database Design Deliverable #1a – Application Business requirements

The Auto Rental Management System's business requirements form the critical foundation for its database design. Database Business Analyst to conduct interviews with EZRental Inc.'s project and business stakeholders, compiling a comprehensive list of essential business data needed for the application. These business requirements serve as the bedrock for the database design process, guiding the Database Analyst/Architect in the initial phase of data modeling. During this phase, the requirements are translated into either a Conceptual Entity-Relational Diagram (ER) or a Conceptual Enhanced Entity-Relational Diagram (ER), both of which are fundamental to structuring the database. The significance of these requirements cannot be overstated, as they directly influence the system's design and functionality, with detailed documentation provided in the subsequent pages, accompanied by a legend explaining the highlighting markings for clarity.

Business Requirements

About Us:

EZ-Car Rental is an auto rental company that rents vehicles such as cars, SUVs, minivans & cargo vans to customers. In addition, other specialized vehicles such as trucks, motorcycles, boats, mobile homes, etc. We operate in several countries with rental agency locations in the US, Canada, Mexico, UK, Japan & Australia. Within each country we own and operate rental agencies located in cities, regions and state. For example, New York City has 2 rental agencies in Manhattan, one in Brooklyn and two in Queens located at each airport. With multiple rental agencies in cities, states etc., a customer can pick up a vehicle in one location and drop it off at another.

Current Challenges:

Our current rental system is outdated, with a poor user-experience, inefficient (breaks often thus expensive to operate), does not meet our business requirements, and is not scalable (cannot be easily updated with new features). Another very important shortcoming of the current system, is the lack of elasticity since it does not give us the flexibility to scale-up or scale-down resources during business trends and seasonal changes in the market.

We want to invest in modernizing our business with a new vehicle management system that can meet these challenges and delivers a great user-experience, meet our new business requirements, scalable, and elastic to adopt to business trends and seasonal market changes. Elasticity is very important since recently we have been faced with a new type of competition; small rental companies that are nimble and can quickly adopt to market changes thus able to provide new offerings that are appealing to customers thus affecting our profits. These smaller competitors are using new technologies that enable them to be nimble and elastic. Figurative speaking "*they are eating our lunch*".

We look forward to your proposed architecture & implementation of this new system. Below are our business requirements.

Our Rental Agencies:

A **rental agency** is the location where customers visit to pick up and drop off rental vehicles. Each **rental agency** is identified by a unique ***rental agency ID*** number, ***agency name***, ***address*** that is composed of the following elements: ***address line1***, ***address line 2 (***which is optional and used for apartment number, suite or any additional address information required), **city**, ***state code*** (which is the two-character code for a state in the US), ***zip code*** & ***country***. In addition, we also need to capture the agency's ***phone number,*** and ***email*** which is unique for all agencies as all emails are.

Our Customers:

EZ-Car Rental offer their services to two types of **Customers**: **Corporate Customers** & **Retail Customers**. **Corporate Customers** are individuals whose corporation have a contract with us to use our services with special corporate rate for their employee's rental services. On the other hand, **Retail Customers** are consumers not associated with a company and engaging in personal rental.

Requirements for All Customers (Retail & Corporate Customers)

To run our business, the application must store the following customer information for both types of **customers** (***retail*** & ***corporate***) so this data is common to both types of customers:

- A ***Customer ID*** number which uniquely identifies the customer, ***customer name*** which is composed of: ***first name***, ***last name***.

- ***Birth date***, ***Age***, ***Address*** which includes the elements: ***address line1, address line 2 (***which is optional and used for apartment number, suite or any additional address information required), ***city***, ***state code*** (which is the two-character code for a state in the US), ***zip code*** & ***country***.
- Customer ***phone number*** & ***email*** (unique like all emails and required to rent).
- In addition, a driver license is required to reserve and rent a vehicle. Therefore, we need to capture the unique ***driver license number (****an alpha numeric character string containing numbers & characters****. Note that in the USA the format for the driver license number can be under 15 characters, but in other countries it could go up to 22 to 25 characters*) ***driver license expiration Date*** and ***driver license state***. In addition, note the following ***business rule*** policy regarding the business importance of the ***driver license number***:

1. ***The driver license number is used throughout the business to identify a customer for searching, reporting etc.***
2. ***Therefore, the driver license number is the main unique ID for a customer to be identified and managed from a business perspective.***

Business Requirements

Our Customers (Cont.):

- A very important attribute we need to capture for every customer is the **credit card**. For our credit card processing and transactions, we need to capture the following ***credit card*** components: ***credit card number*** that uniquely identifies the credit card and is a 16-character number digits. We also need to capture the ***credit card owner name***, in addition, credit card processing attributes such as ***credit card issuing bank code***, ***credit card issuing bank name***, ***credit card network company code***, ***credit card network company name***, ***credit card processing merchant service company code***, ***credit card merchant service company name***, ***credit card corporate merchant bank code***, and ***credit card corporate merchant bank name***. Important – further details on these credit card processing attributes will be provided in sections to follow. Is important that these attributes are clearly understood to correctly design the system.
- Other attributes of credit card are ***expiration date***, ***billing address*** composed of ***address line1, address line 2 (***which is optional and used for apartment number, suite or any additional address information required), ***city***, ***state code*** (which is the two-character code for a state in the US), ***zip code*** & ***country***, ***credit card limit*** (which is the maximum amount of money a customer can charge on their credit card), ***credit card available credit*** (which is how much you have left to spend with your credit card or unused amount within your limit). Note that we will capture the credit card limit to a maximum of $999,999.99, since we don't expect our customers to have a credit limit of $1 Million dollars. Finally, ***activation status*** (which a is true if the credit card is active and can be used, or false when the credit card is not active or disabled).
- During the interview with business stakeholders provided the following ***Business Rules*** related to a credit card:

1. ***You cannot reserve or rent one of our vehicles without a credit card.***
2. ***A customer can have many credit cards they can use to pay for rental transactions***.
3. ***A credit card can be owned by the one customer or co-owned by other individuals such a family member or corporate entity the customer works for. Therefore, many customers can own the same credit card and a credit card can be owned by many customers***.

The Credit Card processing Workflow

Processing of the credit card transactions is a key part of this business and is important that we store data for each step of the credit card processing process. Therefore, the credit card processing attributes discussed in previous section need to be further analyzed and clearly understood. We will now provide the definition and detailed information on each of these credit card processing attributes: ***credit card issuing bank code***, ***credit card issuing bank name***, ***credit card network company code***, ***credit card network company name***, ***credit card processing merchant service company code***, ***credit card merchant service company name***, ***credit card corporate merchant bank code***, and ***credit card corporate merchant bank name***:

- ***Credit Card Processing Merchant Service Company*** – In credit card processing the merchant is the retailer where a customer purchased the goods and services and pay using a credit card. **EZRental Inc**., is the merchant in this scenario. The ***Credit Card Processing Merchant Service Company*** is the institution which works directly with the merchant (**EZRental Inc.**) to provide handle the credit card processing services and handles all the complexity of credit card processing and interactions with the other financial entities involved in the credit card processing process on behalf of the merchant (**EZRental Inc.**). The ***Credit Card Processing Merchant Service Company*** provides the Merchant or Business with the hardware which the customer swipes or inserts to pay for goods and services with their credit card. As part of the credit card processing cycle, the first financial institution which the ***Credit Card Processing Merchant Service Company*** interacts with is the ***Credit Card Network Company*** (which we will cover next)**.** The ***Credit Card Processing Merchant Service Company*** ensure the merchant (**EZRental Inc.**) is connected to the right ***Credit Card Network Company***. We will describe the ***Credit Card Network Company*** next, nevertheless, we need to capture the following information for the ***Credit Card Processing Merchant Service Company***:

- ***Credit Card Processing Merchant Service Company Code*** – In our business, we use and store a number code used to identify the *Credit Card Processing Merchant Service Company*. This code has business meaning and appears in reports and discussed by users. Therefore, a *Credit Card Processing Merchant Service Company* can be identified by this code.
- ***Credit Card Processing Merchant Service Company Name*** – In our business, we also use and store the name of the ***Credit Card Processing Merchant Service Company***. This name also has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Processing Merchant Service Company*** can be identified by its name.
- Below is a listing of the values/instances of the ***Credit Card Processing Merchant Service Company Code*** and ***Credit Card Processing Merchant Service Company Name*** we use at **EZRental Inc**:

Business Requirements

Our Customers & Credit Card Processing (Cont.):

- ***Credit Card Network Company*** – In credit card processing the objectives of the ***Credit Card Network Company*** is to process transactions between the ***Credit Card Issuing Bank*** (which we will cover next) and the ***Credit Card Processing Merchant Service Company***. Covered previously. The ***Credit Card Network Company*** act like bridges between the ***Credit Card Issuing Bank*** that issue credit card and the ***Credit Card Processing Merchant Service Company*** that handles the transaction from the merchant (**EZRental Inc.**). The ***Credit Card Network Company*** that interacts with the ***Credit Card Issuing Bank*** to determine whether to approve or deny the transaction. And then the ***Credit Card Network Company*** notifies the merchant (**EZRental Inc.**) if the purchase was approved or denied. The ***Credit Card Network Company*** is a digital infrastructure that facilitates credit card transactions and prepares the transaction for the ***Credit Card Issuing Bank***. We will describe the ***Credit Card Issuing Bank*** next, nevertheless, we need to capture the following information for the ***Credit Card Issuing Bank***:

  - ***Credit Card Network Company Code*** – We use and store a number code to identify the ***Credit Card Network Company***. This code has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Network Company*** can be identified by this code.
  - ***Credit Card Network Company Name*** – In our business, we also use and store the name of the ***Credit Card Network Company***. This name also has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Network Company*** can be identified by its name.
  - Below is a listing of the values/instances of the ***Credit Card Network Company Code*** and ***Credit Card Network Company Name*** we use at **EZRental Inc.**:

- ***Credit Card Issuing Bank*** – The ***Credit Card Issuing Bank*** is the financial or lending institution that offers the Credit Card and pays for the goods and services until the customer pays back the credit/loan. These are Banks, Lending Institutions, Credit Unions, Fintech companies, etc. These institutions issues/provides the credit for the customer. The cardholder borrows money from the credit card issuing bank each time they make a purchase, and when they pay their credit card bill, they're paying the ***Credit Card Issuing Bank***. We need to capture the following information for the ***Credit Card Issuing Bank***:

- ***Credit Card Issuing Bank Code*** – In our business, we use and store a number code used to identify the ***Credit Card Issuing Bank***. This code has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Issuing Bank*** can be identified by this code.
- ***Credit Card Issuing Bank Name*** – In our business, we also use and store the name of the ***Credit Card Issuing Bank***. This name also has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Issuing Bank*** can be identified by its name.
- Below is a listing of the values/instances of the ***Credit Card Issuing Bank Code*** and ***Credit Card Issuing Bank Name*** we use at **EZRental Inc.**:

Business Requirements

Our Customers & Credit Card Processing (Cont.):

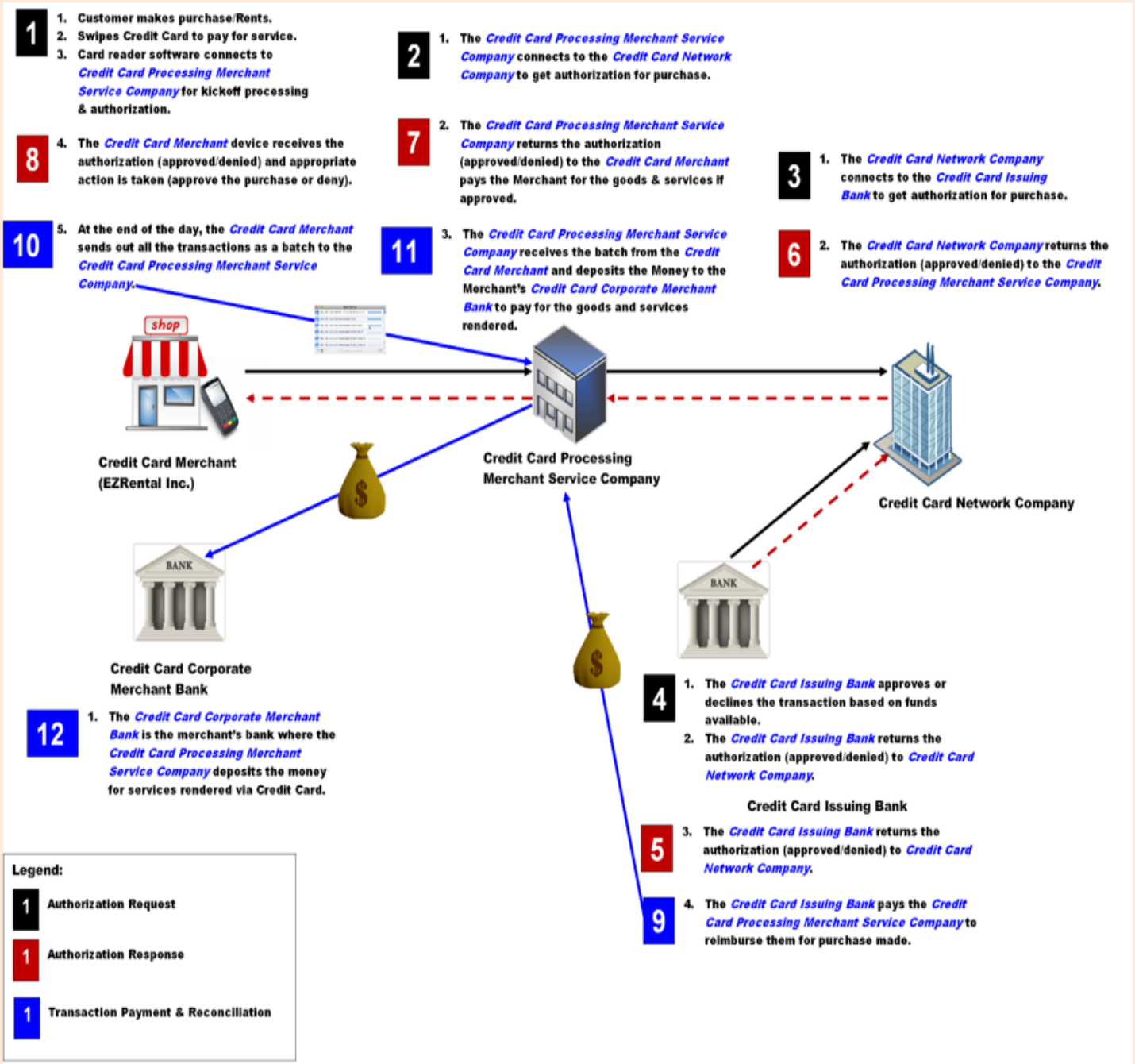
- ***Credit Card Corporate Merchant Bank*** – In credit card processing, the ***Credit Card Corporate Merchant Bank*** is the bank used by the merchant **EZRental Inc.**, to handle credit card processing money transactions, payments, etc., between **EZRental Inc.,** and the ***Credit Card Processing Merchant Service Company*** that handles the Credit Card Processing on behalf of **EZRental Inc**. In short, it is the bank that has the bank account used by **EZRental Inc.,** to handler the accounting for Credit Card Processing Transactions. We need to capture the following information for the ***Credit Card Corporate Merchant Bank***:

  - ***Credit Card Corporate Merchant Bank Code*** – In our business, we use and store a number code used to identify the ***Credit Card Corporate Merchant Bank***. This code has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Corporate Merchant Bank*** can be identified by this code.
  - ***Credit Card Corporate Merchant Bank Name*** – In our business, we also use and store the name of the ***Credit Card Corporate Merchant Bank***. This name also has business meaning and appears in reports and discussed by users. Therefore, a ***Credit Card Corporate Merchant Bank*** can be identified by its name.
  - Below is a listing of the values/instances of the ***Credit Card Corporate Merchant Bank Code*** and ***Credit Card Corporate Merchant Bank Name*** we use at **EZRental Inc.**:

Business Requirements

Our Customers & Credit Card Processing (Cont.):

Below, is a pictorial representation of the interaction between the credit card processing entities *Merchant (EZRental Inc.)*, *credit card processing merchant service company*, *credit card network company* and *credit card issuing bank*:



In summary, we need to capture the data for the credit card processing attributes: *credit card issuing bank code*, *credit card issuing bank name*, *credit card network company code*, *credit card network company name*, *credit card processing merchant service company code*, *credit card merchant service company name*, *credit card corporate merchant bank code*, and *credit card corporate merchant bank name*.

Business Requirements

Our Customers (Cont.):

Corporate Customers

Corporate Customers are customers who are renting vehicle during business travel and their company have a contract with **EZRental Inc**. These companies get special corporate rate for their employee's rental services. Therefore, for our **corporate customers only**, we must store the following attributes/properties: unique ***company ID*** (we have a unique ID number for each company doing business with us), ***company name***, ***company address*** which contains the elements: ***address line1***, ***address line 2*** (which is optional and used for apartment number, suite or any additional address information required), ***city***, ***state code***, ***zip code*** (which is the two-character code for a state in the US) & ***country***, in addition, ***company contact*** which is composed of ***company representative name***, ***contact phone number*** & ***contact email*** (unique as all email addresses). And finally, we need to store the ***company discount percentage rate*** which is the discounted percentage applied to a corporate customers rental. The company Discount percentage rate is store in the database as a decimal percentage value, for example 20% is stored as 0.20, 30% as 0.30. 50% as 0.50 etc. This discount percentage (0.0x) is applied to the **Vehicle Rental Categories** which determines the price of each category to determine the total discount. Therefore, when a corporate customer rents a vehicle from a vehicle category (such as economic, compact, standard etc.), this discount percentage is applied to each of the categories during the rental/reservation process. Note that every company has a different percentage rating depending on their contract with **EZ-Rentals Inc**. For example, some companies have 20% discount towards their rentals, which would be stored as 0.20 in the database, some have 30% (0.30) etc. Vehicle Rental Categories are discussed in more details later in these requirements.

Retail Customers

Retail Customer Discounts

Retail Customers can (but don't have to) leverage promotional **discounts** or multiple coupons obtain from other businesses, internet, magazine, organizations, etc., to save money on their rentals. Therefore, we need to capture specific data for the promotional discounts used by a retail customer. A ***Promotional Discount*** is composed of the following attributes: ***discount ID***, a unique random number which uniquely identifies a discount, another unique ***discount code*** or the coupon code itself used to redeem the coupon, which is an alphanumeric code **10-characters** long. This code is generated by our marketing team and published to magazines, newspapers, internet e-commerce sites, etc. Finally, the last attribute is ***discount code description*** or description of the discount. Examples of currently used ***discount ID***, ***discount code***, ***discount code description*** are shown in table below:

Discount ID	*Discount Code*	*Discount Code Description*
1234..	AAA9970054	**AAA Membership Discount - 25% off base rate plus 10% donated for breast cancer research.**
5678..	GOV8756921	**Government Employee Discount - 30% off base rate**
9101..	*STA3415632*	**State Employee Discount for 25% off base rate**
1213..	VET2055179	**Veteran Discount 35% off base rate Plus 10% donation to veteran's family fund.**
Etc..	Etc..	**Etc..**

The following business rules were identified regarding the retail customer discount program:

1. *Only a Retail Customer can use Discounts, no other type of customer, e.g., Corporate Customer, can apply a discount.*
2. *Discounts are applied during reservation of a vehicle or during the actual rental process only.*
3. *A Retail Customer can apply multiple Discounts throughout their lifetime as an EZRental Inc., customer, NEVERTHELESS, ONLY ONE Discount can be applied for a reservation or rental instance. You cannot apply multiple discounts to a reservation or rental.*
4. *A Discount can be used by many Retail Customers and many Retail Customers can use a Discount.*
5. *When a Discount is used by a Retail Customer, we need to capture the Discount Submitted Date which is the date the customer provided the discount and the Discount Redeemed Date which is the date the customer used the discount for a rental. In this business a customer can submit a discount during registration on one date but use it in a future date when they are renting.*

Business Requirements

Our Customers (Cont.):

Retail Customer EZPlus Rewards Program

Retail customers can opt-in to enrolled in the **EZPlus Rewards Program** where they earn points for every rental of a vehicle. These rewards points can be redeemed for future rentals. Note that the **EZPlus Rewards Program** is **optional** for retail customers & points are earned only when they rent vehicles. For the **EZPlus Rewards Program** we need to store unique random number ***EZPlus ID***, the unique ***Ezplus rewards code*** which is the code used in the business when managing the **EZPlus Rewards Program**. This random code is generated and assigned to a **Retail Customer** by the client application. The number starts with the 3-characters EZP and a 10-digit number e.g., **EZP9999999999**, and the final attribute is the ***EZPlus rewards earned points***, which is an integer that indicates the number of rewards points earned that accumulated after all the rentals and can be used to save on future rentals. **Examples of currently used *EZPlus ID*, *EZPlus rewards Code* and *EZPlus earned points* that we currently use are**:

EZPlus ID	*EZPlus Rewards Code*	*EZPlus Rewards Earned Points*
1234..	EZP9009854637	**10000**
5678..	EZP1000192461	**500**
9101..	EZP6493238865	**159000**
1213..	EZP2005135627	**23000**
	Etc..	**Etc..**

The following business rules apply to the EZPlus Rewards Program:

1. ***Only a Retail Customer can leverage the EZPlus Rewards Program, no other type of customer such as a Corporate Customer can join the EZPlus program.***
2. ***The EZPlus Rewards Program is OPTIONAL. A Retail Customer can join the EZPlus Rewards Program during registration or any other time after or not join at all.***
3. ***A Retail Customer can drop out of the EZPlus Rewards program at any time.***
4. ***Every time a Retail Customer that is member of the EZPlus Rewards Program rents a vehicle, they earn 1000 EZPlus Rewards Points. When a Retail Customer member of the EZPlus Rewards Program earns 10,000 EZPlus Rewards Points they earn a FREE RENTAL.***

As an incentive for our retail customers to join the **EZPlus Rewards Program** during registration to become a customer, we offer a ***EZPLus sign-up rewards points*** of **1000** ***EZPlus Rewards Earned Points***. Also note that the maximum number of ***EZPlus Rewards Earned Points*** is capped at **50,000** points. A Retail Customer cannot accumulate more than **50,000** points in the ***EZPlus Rewards*** program.

Business Requirements (Cont.)

Our Customers (Cont.):

In this business, we have the following business rules for our customers (***Retail*** or ***Corporate***):

1. ***We only have two types of customers retail customer or corporate customers. No other type of customer exists.***
2. ***The minimum age to be a customer and rental our vehicles is 21 years old. This rule must be enforced.***
3. ***A customer cannot be a retail & corporate customer at the same time. A customer can only rent as a retail customer or as a corporate and these transactions must be separate. We don't want our customers to be able to combine both retail customer discounts, rewards program and corporate rates at the same time.***

Our Vehicles:

EZ-Car Rental needs a system to manage their vehicles for renting, maintenance, selling, etc. Vehicles are classified into 4 main types: **CAR**, **SUV**, **MINIVAN**, and **CARGO VAN**. These are the vehicles most rented and available at every rental agency. Nevertheless, there are other categories of vehicles available only certain rental agency locations such as **RECREATIONAL VEHICLES**, **MOTORCYCLES**, **MOBILE HOMES**, etc. No matter what type of vehicle being rented, all vehicle types share the following common characteristics:

- Each vehicle is identified by the random number ***vehicle ID***. In addition, each vehicle is also identified by the alpha-numeric ***vehicle VIN number***. Note the following business rule on a ***vehicle VIN number***:

  1. ***The vehicle VIN number is used throughout the business to identify a vehicle for searching, reporting etc.***
  2. ***Therefore, the vehicle VIN number is the unique ID for a vehicle to be identified and managed from a business perspective.***

- Other attributes include the ***vehicle name*** composed of ***make, model*** & **year**. Additional attributes are ***color***, also the ***license plate*** composed of the following components: ***license plate number***, ***license plate state***.
- More attributes are ***mileage***, ***transmission type*** of the vehicle. The Transmission Type attribute has business value thus used in reports and in the business processes. The values used for ***transmission type*** and a ***transmission type description*** as follows:

Transmission Type	*Transmission Type Description*
1	**Manual Transmission**
2	**Automatic Transmission**
3	**Continuously Variable Transmission (e.g., CVT).**
4	**Semi-automatic Transmission**
5	**Dual-clutch Transmission**
6	**Transaxle Transmission**

- ***seat capacity*** attribute, which is the number of seats in the vehicle. Vehicles such as *cars* have a seat capacity of 5 passengers (2 in front and 3 in the back), *SUVs* have 7 or 8 passengers. Cargo Vans have only 2 passenger seat capacity, Minivan have 8 to 9 passengers, special vehicles such as passenger van hold 12 passenger seat capacity, a shuttles bus can hold 16 to 20 passengers, mini-buses 30 to 40 passengers and large busses can hold 70 passengers.

- All vehicles also have a special code and description that we use to track the vehicle status named ***vehicle status ID***. This is a unique number that identifies the status of a vehicle, which works in conjunction with ***vehicle status description*** which describes the status represented by the ***Vehicle Status ID***, such as **reserved**, **rented**, **available**, **maintenance**, **not available**, **transferred**, etc. Below Is the list of vehicle status IDs we are currently using and their descriptions:

Vehicle Status ID	*Vehicle Status Description*
1	**Available**
2	**Reserved**
3	**Rented**
4	**Not available**
5	**Maintenance (Not available)**
6	**Dropped off and located at another agency**
7	**In Transport to Owning Agency**
8	**No Longer available for rental**

Business Requirements (Cont.)

Our Vehicles (Cont.):

In addition to these attributes shared by all vehicles, there are 4 main categories of vehicle which share unique characteristics than the other types of vehicles found in our agencies. These 4 types are as follows:

- A **Car** is a vehicle whose ***trunk capacity*** (measured in cubic feet volume) is advertised to our customers. Customers can decide which vehicles better fits their needs based on the trunk capacity and number of luggage they are carrying etc. For example, a *luxury Mercedes E class* car has a trunk capacity of 18.5 cubic ft., which has a large trunk capacity.
- An **SUV** is a vehicle with a ***towing capacity*** attribute in pounds. Towing capacity is a single number in pound or could also be a decimal number in pounds. For example, some of our SUV have a maximum towing capacity of 3,000 pounds etc. Another attribute of SUV is an attribute classification if the SUV is ***All-Wheel-Drive***, which stores a Boolean value of **YES/NO** or **TRUE/FALSE**.
- A **Minivan** has the option of ***having a disability package***, which is also a Boolean value of **YES/NO** or **TRUE/FALSE**.
- Finally, a **Cargo Van**, has a ***cargo capacity*** in cubic feet volume. For example, the typical volume of our Vans is 245 cubic feet (cu.ft.). Cargo Vans also have a ***maximum payload*** attribute that determines how much weight in pound it can hold. Our cargo vans have typically a maximum payload of 3,880 lbs.

- As stated previously, there are other types of vehicles of interest that in some location we may want to store data on other than car, SUV minivans and cargo van.
- Note that the following Business Rules were identified by the business stakeholders on the vehicles:

1. ***A reservation/rental can only be for one of these four categories of Vehicles or other vehicle types, not a combination.***

2. ***This means, you can only rent either a car, SUV minivans, cargo van or other for a reservation or rental, not a combination such as a car & SUV at the same time. Each reservation is unique to one vehicle.***

Below are additional business rules for our vehicles and agency ownership:

1. ***Every vehicle is owned by one agency. The vehicle can be pick-up and dropped-off at any agency, but only one agency is the vehicle's owning agency. An agency can own many vehicles, but a vehicle can only be owned by one agency.***

2. ***A vehicle can currently be located at any agency depending on where it was dropped-off after a rental. We need to track the current agency where the vehicle is located, to arrange a transfer or a rental that will ultimately direct the vehicle to the owning agency.***

Reservation Process:

A vehicle must be reserved if a customer wants to guarantee the vehicle will be available for rental. There is a distinction between a reservation and a rental. A reservation guarantees a vehicle will be ready for you to be pick-up and rented. A rental means a customer complied with the reservation and rented the vehicle. On the other hand, a customer can walk into an agency and rent without reservation but only vehicles that are available at the time and not reserved.

We have the following business rules for reserving a vehicle reservation:

1. ***A reservation is NOT made for a specific vehicle, but to a vehicle rental category. Rental category examples are economy, intermediate, full size, luxury.***

2. ***Thus, a customer makes a reservation of a vehicle rental category at a rental agency. Therefore, the reservation process involves a customer a vehicle rental category and the rental agency*** where the vehicle will be picked up.

Business Requirements (Cont.)

Reservation Process (Cont.):

A **Vehicle Rental Category** contains a list of vehicles depending on the vehicle type: Car (economy, intermediate, full size, luxury), SUV (standard, full size etc.), or Cargo Van etc. Each of these categories have a different price range. Therefore, for a vehicle rental category we need to capture the unique ***vehicle rental category ID*** that identifies the category of the vehicle being reserved or rented, ***category name*** and finally ***category daily rental rate*** for the category. We used a specific code for our vehicle rental category ID, category name & daily rental rate. The table below shows the ID, category names and rate we currently using in our business:

Vehicle Rental Category ID	*Vehicle Rental Category Name*	*Category Daily Rental Rate*
1	Car-Economic	$113.99
2	Car-Compact	$115.99
3	Car-Intermediate	$116.67
4	Car-Standard	$119.99
5	Car-Full Size	$121.99
6	Car-Premium	$127.79
7	Car-Luxury	$139.99
8	SUV-Intermediate	$127.99
9	SUV-Standard	$128.99
10	SUV-Standard Elite	$135.99
11	SUV-Full Size	$148.99
12	SUV-Premium	$157.99
13	Minivan-Standard	$152.99
14	Van-Cargo Van	$19.95
15	Pick Up-Mid Size	$69.95
16	Pick Up-Full Size	$105.99
17	Motorcycle-Touring	$19.95
18	Motorcycle-Cruiser	$199.99
19	Motorcycle-Scooter	$79.95
20	Passenger Van (12 passengers)	$161.00
21	Passenger Shuttle (16 passengers)	$180.00
22	Passenger Shuttle (20 passengers)	$220.00
23	Passenger Mini-Bus (30 passengers)	$250.00
24	Passenger Mini-Bus (40 passengers)	$280.00
25	Passenger Large-Bus (80 passengers)	$300.00

We have the following business rule relate to a vehicle and a vehicle rental category:

1. *A vehicle is a member of a vehicle rental category.*
2. *A vehicle rental category can have one, none or many vehicles belonging to that category at any given time, nevertheless, a vehicle can only belong to one vehicle rental category.*

As stated previously, **a customer makes a reservation of a vehicle rental category at a rental agency**. Therefore, the reservation process requires the **customer**, **vehicle rental category** & **rental agency** for a reservation to be made. The following business rules apply to a reservation:

1. *A vehicle can be reserved to be picked up at the **INDICATED** rental agency and dropped off at the **SAME** rental agency.*
2. *A vehicle can be reserved to be picked up at the **INDICATED** rental agency and dropped off at a **DIFFERENT** rental agency.*
3. *A reservation is made only for one pick-up rental agency, but a rental agency can have many reservations for pick-ups taking place.*
4. *A reservation can only be for one drop-off rental agency, but a rental agency can have many reservations drop-offs taking place.*
5. *For reporting and analytics, we need to capture all changes to the reservation's pick-up and drop-off agency, such as all changes by the customer for pick-up agency & drop-off agency later after the original reservation. This means we need to store all history on all reservation pick-and drop-off agency changes.*

When a customer reserves a vehicle rental category for a specific rental agency, we wish to capture the following:

- A unique ***reservation ID*** which is used by the business to manage and track reservations, the ***rental agency ID*** where the vehicle will be picked up, and the target ***reservation drop-off rental agency***.
- In addition, we need to store the ***reservation schedule*** composed of ***reservation pick up date***, ***reservation pick up time***, ***reservation drop off date*** and ***reservation drop off time***, also the ***reservation estimated rental cost***. A reservation can have multiple schedule changes during the lifetime of the reservation since customers can make changes to the reservation and we need to track this history of changes for analytical purposes.

Business Requirements (Cont.)

Reservation Process (Cont.):

We have the following business rule relate to the ***reservation schedule***:

1. ***For reporting and analytics, we need to capture all changes to the reservation schedule, such as all changes by the customer for pick-up date & time and drop-off date & time. This means we need to store all history on all reservation schedule changes.***
2. ***A customer can have MANY reservation schedules based on changes to the reservation, but a schedule can only belong to ONE customer.***

- Finally, we need to store the unique ***reservation status ID*** which is a unique number we use to indicate the status of a reservation and ***reservation status description*** which describe each of the status such as: **confirmed**, **cancelled**, **completed** etc. Below is an example of the ***reservation status ID*** and ***status description*** we currently use in our business.

Reservation Status ID	*Reservation Status Description*
1	Confirmed
2	Modified & reconfirmed
3	Cancelled
4	Fulfilled & closed
Etc..	Etc..

For a reservation we must adhere to the following business rules:

1. ***A customer can make none, one or many reservations for a vehicle rental category at a rental agency.***
2. ***A rental category can be reserved by none, one or many customers at a rental agency.***
3. ***A rental agency can get many or no reservations for a vehicle rental category by a customer.***
4. ***A reservation can only have one pick-up rental agency location, but a rental agency can have many reservation pick-ups happening.***
5. ***Each reservation has a drop-off rental agency (may be different than pick-up rental agency). A reservation can only have one drop-off rental agency location, but a rental agency can have many reservation drop-offs taking place.***

The Rental Process:

Once a vehicle has been reserved, the vehicle can be rented (picked up/dropped off) as per the scheduled of the reservation agreement. **A rental means a customer complied and fulfilled the reservation and rented the vehicle**.

For the rental process, the following business rules apply:

1. ***A customer rents a vehicle Rental Category at a rental agency. This means the rental process requires the customer, vehicle rental category, and & rental agency for a rental to be complete.***
2. ***A Rental includes a specific Vehicle of the vehicle rental category. A vehicle can be rented many times, but a rental is only for one vehicle only. You cannot rent multiple vehicles in one rental contract.***
3. ***During the rental process we may have any of the following business rules/scenarios:***

    1) ***A vehicle can be picked up at the SAME rental agency as indicated by the reservation and dropped off at the SAME rental agency.***
    2) ***Or a vehicle can be picked up at the SAME rental agency as indicated by the reservation and dropped off at ANOTHER rental agency.***
    3) ***Or a vehicle can be picked up at ANOTHER rental agency other than what was indicated by the reservation and dropped off at SAME rental agency of the reservation.***
    4) ***A vehicle can be picked up at ANOTHER rental agency other than what was indicated by the reservation and dropped off at ANOTHER rental agency of the reservation.***
    5) ***For reporting and analytics, we need to capture all changes to the rental pick-up and drop-off agency, such as all changes by the customer for pick-up agency & drop-off agency later after the original reservation. This means we need to store all history on all rental pick-and drop-off agency changes.***

    - Note that for scenarios 3 & 4, we cannot guarantee that the vehicle rental category of the reservation will be available at the agency other than what was agreed in the reservation. We will do our best to accommodate the change during these scenarios or find another vehicle that will be closed to the original reservation.

For the rental process, the following business rules also apply:

1. ***A rental can only be for one pick-up rental agency, but a rental agency can have many rental pick-ups taking place.***
2. ***A rental can only be to one drop-off rental agency, but a rental agency can have many rental drop-offs taking place.***

When a customer rents a vehicle at the rental agency, we need to capture the following information about the rental:

- The ***rental agreement ID*** that uniquely identifies the rental transaction, ***rental pick up date***, ***rental pick up time***, ***rental drop off date*** and ***rental drop off time***, ***rental pick up odometer value*** and ***rental drop off odometer value***.

Business Requirements (Cont.)

The Rental Process (Cont.):

- In addition, a customers receive a vehicle with a full tank of gas and customers are expected to return the car on a full tank of gas otherwise they must pay a penalty upon return. Since we understand our customers are busy and may forget to return the car with a full tank of gas, we offer our customers with the option to pay in advance for a full tank of gas at our rates and don't have to worry about returning the vehicle with a full tank of gas. Therefore, we need to capture the unique ***rental fuel option ID*** or option chosen by the customer, ***rental fuel option description*** and ***rental fuel option additional cost***. We currently use the following fuel option IDs, descriptions, and example of each of the additional cost for the fuel option:

Rental Fuel Option ID	*Rental Fuel Option Description*	*Rental Fuel Option Additional Cost*
1	Return with a full tank or on return, pay for gas that is missing.	$13.97 (Important, this Decimal value of $13.97 is just an example, since the value is calculated during car return process and is based on the current price for a gallon of gas etc. therefore price will vary.)
2	Pay for full tank in advanced at time of rental, return car empty. No refund for unused gas.	$45.99 (Important, this Decimal value of $45.99 is just an example, since the value is calculated during car return process and is based on the current price for a gallon of gas etc. therefore price will vary.)

- Also, we give customer options for car insurance & protection, therefore we need to capture the unique ***insurance option ID***, ***insurance option description*** and ***insurance option additional cost***. We currently use the following insurance option IDs, descriptions, and cost:

Rental Insurance Option ID	*Rental Insurance Option Description*	*Rental Insurance Option Additional Cost per Day*
1	No insurance. Opt-out.	$0.00
2	Collision Damage Waiver Max - Agency will pay for damage, lost or stolen vehicle.	$49.99
3	Collision Damage Waiver 3000 - Agency will pay for first $3,000 of loss or damage, renter pays all loss & damage after $3,000.	$39.99
4	Lability Extended Protection – Agency provides renter with third party liability protection up to $1 Million per accident for bodily injury or death or property damage to others.	$89.99
5	Roadside Assistance Plus – 24/7 roadside assistance, replacement for lost keys, flat tire service, fuel delivery, etc.	$15.99

- Other attributes required for the rental that we need to capture are the unique ***rental status ID*** & ***rental status description***. We currently use the following rental status IDs & descriptions:

Rental Status ID	*Rental Status Description*
1	Picked up as scheduled.
2	Dropped off as scheduled.
3	Returned late
4	In progress.
5	Roadside assistance in progress.
7	Unknown

Business Requirements (Cont.)

The Rental Process (Cont.):

- Other attribute we need to capture the ***rental deposit*** for a rental. The rental deposit value is calculated based on the **rental period + 25% of the rental period** and for any damage or other charges that were incurred during the rental period. This deposit is refunded to the customer's credit card when the vehicle is returned in the condition in which it was rented.
- Finally another attribute we need to capture is the ***rental total cost*** or total cost that needs to be paid by the customer. This value is calculated based on selected *fuel option*, *insurance option*, *vehicle rental category* price and other factor such as such as duration of the rental etc.

We need to be able to associate a reservation to a rental and vice versa, therefore we maintain the following additional business rules for our rental & reservation:

1. ***A reservation is made for a rental and the opposite holds true; a rental is based on a reservation.***
2. ***But NOT all rentals are based on a reservation. We allow a customer to walk into a rental agency and rent a vehicle without a reservation.***
3. ***When a reservation is made for a rental, then it must be for only one rental, and a rental can be for a reservation but not mandatory since a customer can walk into an agency and rent a vehicle without a reservation.***

Our Employees:

EZ-Car Rental currently has 5,500 employees across the world. We do expect to grow as we move into new markets such as Asia, Africa, and the Mediterranean. But our business does not require a large workforce, therefore, we don't expect to grow more than 12,000 in the next 10+ years. Our employees consist of *customer service agents* in the Rental Agencies & online support who interact with our customer to reserve and rent vehicles. In addition, *back-office inventory personnel*, *auto specialists* who work in our services centers servicing our vehicles, *drivers* to transport our vehicles from one agency to another and *maintenance personnel* who maintain our agencies and finally our *business team* that handles the day-to-day business activities in our agencies and other roles. For now, we are only interested in storing the following data for all these types of employees:

- An ***Employee ID*** which uniquely identifies the employee, ***employee name*** which is composed of: ***first name***, ***last name***, also ***employee address*** which includes the components: ***address line1***, ***address line 2***, ***city***, ***state code***, ***zip code*** & ***country***. Also, ***employee phone***, ***employee job title*** and ***employee email***. In addition, we need to capture the employee ***social security number***. Below are some business rules and usage for the ***EmployeeID*** and the ***social security number***.

The following business rules related to employees must be followed:

1. ***The employee social security number needs to be protected and secured as per federal regulations. All security measures such as encryption, etc., need to be taken to protect the social security number; therefore, the full social security number cannot be seen by employees, reports, and other business processes.***
2. ***In special cases where the social security number needs to be displayed, only the last 4 digits will be shown using the following format ****_**_1234. Nevertheless, the goal is NOT to display the social security number as much as possible, and it should only be used internally within the application for processing but not displaying.***
3. ***The EmployeeID number is what is used throughout the business to identify an employee for searching, reporting. business processing, etc., therefore, the EmployeeID is the unique ID for an employee to be identified and managed from a business perspective.***
4. ***The minimum age to be an employee of our company 18 years old. This rule must be enforced.***

Security & Application Access:

To access our systems proper security and authentication is required. Only authorized users can have access our agencies Point-0f-Sales & Back-End Management systems. In addition to our **EZRental.com** portal by our customers. Therefore, due to security and regulatory compliance purpose, we want to separate the employee access data from the customer access data by using two separate user accounts:

- Employee user accounts
- Customer user accounts

Security Access for Employees to Computer Systems in our Agencies (Employee User Accounts):
For our authorized employees & customer service employees to access the agencies Point-0f-Sales & Back-End Management systems they need to log in by entering a username & password for access to the application. This means every employee owns an employee user account.

An employee user account should store the user ***employee user account ID*** a unique identifier alpha-numeric string that identifies the employee user account, ***employee username*** another unique alpha-numeric that identifies each individual user, the ***employee password*** alpha-numeric that is known only to the user, and finally the employee ***email*** to map the user-account to an Employee. Note the following business rule:

1. An employee can own one employee user account only, and an employee user account can only be owned by one employee only since the user account represents the identify of that one employee.

Business Requirements (Cont.)

Security Access for our Customers who register for our EZ-CarRental.com web site (Customer User Accounts):
Customer who accesses our online portal to reserve and rent our vehicles also need a username and password to access our system, therefore each customer owns a customer user account.

A customer user account should store the user ***customer user account ID*** a unique alpha-numeric string identifier that identifies the customer user account, ***customer username*** another unique alpha-numeric value that identifies each customer, the ***customer password*** that is an alpha-numeric known only to the customer, and finally, the customer ***email*** to map the customer user-account to a customer. Note the following business rule:

1. A customer can own one customer user account only, and a customer user account can only be owned by one customer.
2. For a period of time, we will need to register customers into our **EZRental.com** business, nevertheless the web portal may NOT be implemented or completed when new customers are registering at this time, therefore, for period of time, creating a customer user account when registering a new customer is optional until the Web Portal Application is created. But is important in the future, that we force the creation of customer user accounts when a new customer is registered once the Web Portal Application is ready. It is the responsibility of the database architect(s) and full-stack developers to update this feature when the appropriate time comes.

Vehicle Transportation:

We need to know where our vehicles are located at all times, such as at the Rental Agency that owns the vehicle, another Rental Agency that does not own the vehicle, being transported from one Rental Agency to another as a result of a vehicle transfer after a rental to the owning rental agency, being transported as a new delivery to a Rental Agency from our distribution center, being transported for maintenance, or currently being rented by a customer. Vehicles need to be tracked or location status known. At this time, we are only interested in tracking when a vehicle is transported from one Rental Agency to another Rental Agency under the following scenarios:

- Vehicle can be located at a Rental Agency that does not own the vehicle after a rental dropping off at a different location than the picked up owning Rental Agency, thus vehicle eventually needs to be transported and delivered to the owning agency.
- Another non-owning Rental Agency requests support from other Rental Agency(s) for loans of vehicle(s) to borrow due to an unexpected busy period and requesting agency is short on inventory. After the first agency is done with the loaner vehicles, these vehicles need to be returned to the borrowed owning Rental Agency(s).
- In our current process & systems we currently use the following reason IDs and reason descriptions:

Transport Reason ID	*Transport Reason Description*
1	**Rental Drop off at different location**
2	**Vehicle Loaned to another Agency**
3	**Pick up from Distribution Center**
4	**Drop off to Distribution Center**
5	**Vehicle sent for maintenance**
7	**Unknown**

Note that transportation to and from Rental Agency is executed by an employee who is part of a transportation team or drivers. Therefore, when an employee executes a transport request of a vehicle to and from Rental Agencies, we need to capture the following information:

- ***Transport pickup agency ID, Transport drop-off agency ID, Driver departure date, driver departure time, vehicle pick up date, vehicle pick up time, transport completed arrival date, transport completed arrival time, estimated arrival date, estimated arrival time***, & actual ***transport time to completion***.

- In addition, we need to know at any time the transport status and transport status description of the transfer, such as: transfer completed, on route to pick up location, on route from pick up location, etc. Therefore, we need to capture the ***Transport Status ID*** or unique number that identifies a status and the ***Transport Status Description***, or description of each status ID. Currently we track a transportation event using the following ID and description:

Transport Status ID	*Transport Status Description*
1	**Transport completed**
2	**On route to pick up location.**
3	**On route from pick up location**
4	**At pickup location. In progress (Loading etc.)**
5	**Pickup location delay**
7	**Unknown**

The goal again is to be able to know where our vehicles are located at any time and their status.

Business Requirements (Cont.)

Conclusion:

The business data listed in this business requirements document is what we need to capture for our business to operate. As our business evolve, additional data will be required in the future. We will address these new requirements in future versions of the application. For example, invoice processing & employee management at our rental agencies are features on our roadmap. Therefore, our expectations are that the design is modular and scalable for future growth.

5.Database Design Deliverable #1b – Application Development Technical Requirements:

The technical business requirements for the Auto Rental Management System are pivotal to its front-end application design and development, as detailed in the compiled documentation. Application Analyst/Architect to collaborate with EZRental Inc.'s Business Decision Makers (BDMs), stakeholders, and Information System Technical Decision Makers (TDMs) to gather a comprehensive list of Application Development & Technical Requirements essential for designing and building the application. These technical requirements serve as the foundation for the front-end design process, enabling the Object-Oriented-Application Architect and UX/User-Interface Architect to undertake critical tasks such as Object-Oriented Object Model Design, User-Interface Wireframe creation, and UI Design. From these requirements, key deliverables are derived, including the Application Physical Architecture Design, Application Software Design & Programming, an Application Feature List for the Agile Backlog, User-Interface Wireframe Design & Programming, and other related implementations. Ultimately, the technical requirements mark the starting point for the entire front-end application design and development process, ensuring a structured and effective approach.

Application Development & Technical Requirements

Introduction & Current Challenges

As described in the Business Requirements, the current rental system is outdated, with a poor user-experience, breaks often thus expensive to operate, does not meet our business requirements, and is not scalable so it cannot be easily updated with new features etc. Also, not elastic since it does not give us the flexibility to scale-up or scale-down based on business trends and seasonal changes in the market. We want to invest in modernizing our business with a new vehicle management system that can meet these challenges and give us a great user-experience, meet new business requirements, scalable, and elastic to adopt to business trends and seasonal market changes.

We have an outdated IT infrastructure in our datacenter and there is a current initiative to modernize our datacenter and also leverage cloud technology in a hybrid environment to save on cost, streamline our operations and drive innovation.

We look forward to your proposed architecture & implementation of this new system that will meet these requirements. Next sections contain the results of our application development & technical requirements.

Rental Agencies Application & Technical Requirements:

The rental agencies are location where customers both Retail & Corporate will engage our *Customer Service Representatives* to engage in rental/return activities in addition to other transactions such as registering, searching & updating customer information etc. Therefore, the application in the rental agencies is vital to the user-experience for both our *Customer Service Representatives* as well as our *customers*.

We are forecasting that is some locations such as major city centers and airports, there will be many customers engaging throughout the day thus increasing the risk of a poor customer experience in addition to the work overload and poor experience for our *Customer Service Representatives*. We want our *Customers* to be serviced quickly and efficiently with a great experience, and our *Customer Service Representatives* to be able to process each *Customer* easily and effectively. With these criteria in mind, the application at our rental agencies must adhere to the following requirements:

Rental Agency Application Architecture Requirements:

Below are the requirements for the application used in our rental agencies by our customer service representatives, inventory team, service personnel and other employees working in our agencies:

1. Client application processing, transaction and response must be fast to minimize service time for a customer.
2. All transaction processing should be done in the user's computer or desktop for fast processing and response.
3. Application Architecture must be reusable and scalable to support future updates and new feature enhancements, without a long development lifecycle.
4. Depending on the architecture NYC-Tech Solutions Inc., decides for the application in the rental agencies (Desktop client or Web client), the primary Application Development Platform we use is ***C# & .NET technologies***. For any Web related development, we support JavaScript, React, NodeJs and other standard Web Technologies. We have aligned ***C#.NET*** & **ASP.NET Web developers** that have been assigned to assist, support and update the application once NYCTech consultants complete the project and development of this system.
5. Rental Agency Desktop Application Security Authentication System – Proper security and authentication must be implemented to make sure only authorized customer service representative and other rental office employees can access the Point-0f-Sales with appropriate conditional access.

Application Development & Technical Requirements (Cont.)

Rental Agency Application Features and Functionalities Requirements:

The list of features and functionalities that we have compiled for the rental agencies' application are listed in the table below:

No.	Feature	Functionalities
1	**EZRental** Rental Agency Point-of-Sales (POS) System	▪ Car Rental, Car Return, New Customer Registration & Search/Print Customer Information, Customer Update, Customer Deletion, Customer Listing operations etc.
2	**EZRental** Rental Agency Back-Office Vehicle Inventory Management System	▪ Back-office system meant for employees to perform bulk IN-MEMORY inventory processing or management tasks on vehicles such as adding vehicles to the system, searching for vehicles, updating vehicles etc. ▪ This system is NOT meant for Point-of-Sales, but for the inventory management employees who need to search, add, remove etc., a large/bulk number of vehicles or employees during a session. ▪ Back-office vehicle Management features – Allows inventory personnel and employees to bulk-manage Cars, SUVs, Mini-Vans, Cargo Vans to be searched, added, removed, printed, listed etc.
3	**EZRental** Rental Agency Back-Office Credit Card Management System	▪ The EZRental Credit Card Management System is a Back-office system meant for the Credit Card Department Employees to manage Credit Card Information. These uses can Search/Print, Add, Edit & Delete credit card information in the database
4	**EZRental** Rental Agency Back-Office Employee & Customer User Account Management System	▪ The EZRental Customer & Employee User Account Management System is a Back-end system meant for IT ADMINISTRATOR Employees to manage both Employee & Customer USER ACCOUNTS.
5	**EZRental** Rental Agency Desktop Application Security Authentication System	▪ Proper security and authentication must be implemented to make sure only authorized employees can access the Point-0f-Sales, Back-End Management system or any other access to the applications.

Rental Agency Application Graphical User Interface Requirements:

- Graphical User-Interface should be fast rendering and user-friendly workflow.
- Visual screens or forms should be rich in color and appearance and navigation flow should be flexible and easy.
- The following UI controls or data field need to be pre-populated in GUI Screens:

  - **Addresses**

    - Any forms/UI which contains addresses, the STATE & COUNTRY fields should be automatically populated with a list of STATES or COUNTRIES, so the user does not have to manually enter a state or a country and simply select from drop-down list etc.

  - **Discount Codes:**

    - UI screens with customer's DISCOUNT CODE fields should be prepopulated with discount codes. The idea is the user should be able to select the discount to apply to a customer entry from a drop-down list/Combo Box etc. Note that this may or may not include the Discount Code Description on the UI screen as well.
    - Also note that the DISCOUNT CODE VALUES are generated by our Marketing Team and need to be pre-populated in the database before a code can be used. Therefore, the discount codes are prepopulated in the database.
    - Currently, when the Marketing Team generates a new code, they make the request to the database administrator to manually enter an update any new Discount Codes.
    - In the future, we want the application to have the necessary features for the Marketing Team to be able to manage the discount codes. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

Application Development & Technical Requirements

Rental Agency Application Graphical User Interface Requirements (Cont.):

- **EZPlus Rewards Codes:**

  - The EZPlus Reward UI screens with customer's EZPLUS REWARDS CODE fields should be prepopulated with the EZPlus Rewards code for the customer is being applied to. The idea is the user should be able to select the EZPLUS REWARD CODE to apply to a customer entry from a drop-down list/Combo Box etc. or be handled by the back-end database.
  - **Important:** The EZPLUS REWARDS CODE VALUES are NOT generated by a business entity in our organization, but AUTOMATICALLY GENERATED by the application on the fly when registering a new customer. This is a different approach compared to the DISCOUNT CODE which are generated by Marketing Team. In this case, the EZPlus Rewards Code values are generated by the application and available via the UI screen to be used or some other method of generation.
  - To finalize this requirement, the idea is the EZPlus Rewards Code should be automatically generated and either appear in the UI Screen or automatically generated in the database.

- **Company Name:**

  - UI screens with corporate customer's COMPANY NAME fields should be prepopulated with the list of corporations that are members of our corporate program, which enables our users to avoid having to manually enter the company name. Note that this may or may not include the Company ID in the UI Screen which is a unique number with business value that we assign to each company.
  - Note that the company names. Company ids and other company data are managed by our Corporate Sales Team and need to be pre-populated in the database before any corporate customer processing can be made. Therefore, the company information is prepopulated in the database.
  - Currently, when the Corporate Sales Team adds a new corporation or company into the program, they make the request to the database administrator to manually enter and add the new company to the database.
  - In the future we want the application to have the necessary features for the Corporate Sales Team to have the functionality to manage the data of our corporate companies via the application. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

- **Vehicle Status:**

  - UI screens for vehicle inventory management, VEHICLE STATUS field should be prepopulated with the list of vehicle status. Based on the business requirements, the current list of vehicle status is listed in table below:

Vehicle Status ID	*Vehicle Status Description*
1	Reserved.
2	Rented.
3	Available.
4	Not available
5	Maintenance
6	Transferred to another agency

  - Currently populating the database with a vehicle status record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the vehicle status data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

- **Rental Agency:**

  - UI screens that required adding or managing a RENTAL AGENCY field should be prepopulated with the list of rental agencies in our company.
  - Currently populating the database with a rental agency record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the rental agency data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

Application Development & Technical Requirements

Rental Agency Application Graphical User Interface Requirements (Cont.):

- **Vehicle Rental Category:**

  - UI screens that require the use of the VEHICLE RENTAL CATEGORY fields, must be prepopulated with the list of vehicle rental categories. Based on the business requirements, the current list of vehicle rental categories is as follows:

Vehicle Rental Category ID	*Vehicle Rental Category Name*	*Category Daily Rental Rate*
1	Car-Economic	$113.99
2	Car-Compact	$115.99
3	Car-Intermediate	$116.67
4	Car-Standard	$119.99
5	Car-Full Size	$121.99
6	Car-Premium	$127.79
7	Car-Luxury	$139.99
8	SUV-Intermediate	$127.99
9	SUV-Standard	$128.99
10	SUV-Standard Elite	$135.99
11	SUV-Full Size	$148.99
12	SUV-Premium	$157.99
13	Minivan-Standard	$152.99
14	Van-Passenger Van (12 passengers)	$161.00
15	Van-Cargo Van	$19.95
16	Pick Up-Mid Size	$69.95
17	Pick Up-Full Size	$105.99
18	Motorcycle-Touring	$19.95
19	Motorcycle-Cruiser	$199.99
20	Motorcycle-Scooter	$79.95

  - Currently populating the database with vehicle rental category records is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the vehicle rental categories data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

- **Reservation Status:**

  - UI screens that require the use of the RESERVATION STATUS field, must be prepopulated with the list of reservation status data. Based on the business requirements, the current list of reservation status is as follows:

Reservation Status ID	*Reservation Status Description*
1	Confirmed.
2	Modified & reconfirmed.
3	Cancelled & Closed.
4	Fulfilled & Closed.
Etc..	Etc..

  - Currently populating the database with a reservation status record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the reservation status data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

Application Development & Technical Requirements

Rental Agency Application Graphical User Interface Requirements (Cont.):

- **Rental Status:**

  - UI screens that require the use of the RENTAL STATUS field, must be prepopulated with the list of rental status data. Based on the business requirements, the current list of rental status is as follows:

Rental Status ID	*Rental Status Description*
1	Picked up as scheduled.
2	Dropped off as scheduled.
3	Returned late
4	In progress.
5	Roadside assistance in progress.
7	Unknown

  - Currently populating the database with a rental status record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the rental status data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

- **Rental Fuel Option:**

  - UI screens that require the use of the RENTAL FUEL OPTION field, must be prepopulated with the list of rental fuel options data. Based on the business requirements, the current list of rental fuel option is as follows:

Rental Fuel Option ID	*Rental Fuel Option Description*	*Rental Fuel Option Additional Cost*
1	Return with a full tank or on return, pay for gas that is missing.	$13.97 (Important, this Decimal value of $13.97 is just an example, since the value is calculated during car return process and is based on the current price for a gallon of gas etc. therefore price will vary.)
2	Pay for full tank in advanced at time of rental, return car empty. No refund for unused gas.	$45.99 (Important, this Decimal value of $45.99 is just an example, since the value is calculated during car return process and is based on the current price for a gallon of gas etc. therefore price will vary.)

  - Currently populating the database with a rental fuel option record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the rental fuel option data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

Application Development & Technical Requirements

- **Rental Insurance Option:**

  - UI screens that require the use of the RENTAL INSURANCE OPTION field, must be prepopulated with the list of rental insurance options data. Based on the business requirements, the current list of rental insurance option is as follows:

Rental Insurance Option ID	*Rental Insurance Option Description*	*Rental Insurance Option Additional Cost per Day*
1	No insurance. Opt-out.	$0.00
2	Collision Damage Waiver Max - Agency will pay for damage, lost or stolen vehicle.	$49.99
3	Collision Damage Waiver 3000 - Agency will pay for first $3,000 of loss or damage, renter pays all loss & damage after $3,000.	$39.99
4	Lability Extended Protection – Agency provides renter with third party liability protection up to $1 Million per accident for bodily injury or death or property damage to others.	$89.99
5	Roadside Assistance Plus – 24/7 roadside assistance, replacement for lost keys, flat tire service, fuel delivery, etc.	$15.99

  - Currently populating the database with a rental insurance option record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the rental insurance option data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

Application Development & Technical Requirements

- **Transportation Reason Option:**

  - UI screens that require the user to populate the TRANSPORTATION OPTIONS field, must be prepopulated with the list of transportation reason options as shown in the table below:

Transport Reason ID	*Transport Reason Description*
1	Rental Drop off at different location
2	Vehicle Loaned to another Agency
3	Pick up from Distribution Center
4	Drop off to Distribution Center
5	Vehicle sent for maintenance
7	Unknown

  - Currently populating the database with a transportation reason option record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the transportation reason option data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade.

- **Transportation Reason Option:**

  - UI screens that require the user to populate the TRANSPORTATION STATUS field, must be prepopulated with the list of transportation status options as shown in the table below:

Transport Status ID	*Transport Status Description*
1	Transport completed
2	On route to pick up location.
3	On route from pick up location
4	At pickup location. In progress (Loading etc.)
5	Pickup location delay
7	Unknown

  - Currently populating the database with a transportation status option record is handled manually by the database administrator. In the future we would like the application to have the necessary features for our business to be able to manage the transportation status option data. This is not an immediate requirement out of the gate but should be targeted as part of a future upgrade

Application Development & Technical Requirements (Cont.)

Customer Facing Self-Service Web-Portal Application Architecture Requirements:

We now address architecture requirements for the application used in customers via the public internet to make reservations to rent a vehicle, modify their personal account, profile etc.:

1. Customer will use a secure and standard Web Application via a Browser to access our self-service portal in the internet. We need a website to support all customer self-service related transactions.
2. Web Application Architecture must be reusable and scalable to support future updates and new feature enhancements, without a long development lifecycle.
3. For this web development, we support ***JavaScript***, ***React***, ***NodeJS*** and other standard Web Technologies. In addition, the primary Application Development Platform we use is ***C# & .NET technologies***. We have aligned ***C# & .NET*** & **Web** developers that have been assigned to assist, support, operated and update the application once NYCTech consultants complete the project and development of this system.
4. Web Portal Security Authentication System – Proper security and authentication must be implemented to make sure only the customer can access the **EZRental.com** website for his or her profile home page.

Customer Facing Self-Service Web-Portal Features and Functionalities Requirements:

No.	Feature	Functionalities
1	**EZRental.com** Customer Web Portal	▪ Front-end WEB INTERFACE SCREENS & features used by customers via our web portal EZRentalCar.com to reserve a vehicle for rental and manage their account online. ▪ Features include search & reserve a car for rental, register as a new customer, search/view their account information, update their account etc.
2	**EZRental.com** Customer Web Portal Application Security Authentication System	▪ Proper security and authentication must be implemented to make sure only our customer can access the web portal to use the application.

Web Portal Application Web Pages User Interface Requirements:

The web pages graphical UI requirements are listed below:

- The GUI requirements for the web pages are like those functionalities of the Rental Agency Application that are found on the web site for example Search & reserve a car for rental, register as a new customer, search/view their account information, update their account etc.
- The design and graphics of the application should be appealing to customers and a smooth and fluent workflow.
- The following UI controls or data field need to be pre-populated in GUI Screens:

  - **Addresses**

    - Any web-page UI which contains addresses, the STATE & COUNTRY fields should be automatically populated with a list of STATES or COUNTRIES, so the user does not have to manually enter a state or a country and simply select from drop-down list etc.

  - **Discount Codes:**

    - Web pages with customer's DISCOUNT CODE fields should be a text box that allows the customer to ADD/APPLY the discount codes to redeem the coupon.

Application Development & Technical Requirements

Rental Agency Application Graphical User Interface Requirements (Cont.):

- **EZPlus Rewards Codes:**
  - The EZPlus Reward web page screens with customer's EZPLUS REWARDS CODE fields should be prepopulated with the EZPlus Rewards code for the customer is being applied to. The idea is the user should be able to select the EZPLUS REWARD CODE to apply to a customer entry from a drop-down list/Combo Box etc. or be handled by the back-end database.
  - **Important:** The EZPLUS REWARDS CODE VALUES are NOT generated by a business entity in our organization, but AUTOMATICALLY GENERATED by the application on the fly when registering a new customer. The EZPlus Rewards Code values are generated by the application and available via the UI screen to be used or some other method of generation.
  - To finalize this requirement, the idea is the EZPlus Rewards Code should be automatically generated and either appear in the UI Screen or automatically generated in the database.

- **Rental Agency:**
  - Web pages that required adding a RENTAL AGENCY field should be prepopulated with the list of rental agencies in our company.

- **Vehicle Rental Category:**
  - Web pages that require the use of the VEHICLE RENTAL CATEGORY fields, must be prepopulated with the list of vehicle rental categories. Based on the business requirements, the current list of vehicle rental categories is as follows:

Vehicle Rental Category ID	*Vehicle Rental Category Name*	*Category Daily Rental Rate*
1	Car-Economic	$113.99
2	Car-Compact	$115.99
3	Car-Intermediate	$116.67
4	Car-Standard	$119.99
5	Car-Full Size	$121.99
6	Car-Premium	$127.79
7	Car-Luxury	$139.99
8	SUV-Intermediate	$127.99
9	SUV-Standard	$128.99
10	SUV-Standard Elite	$135.99
11	SUV-Full Size	$148.99
12	SUV-Premium	$157.99
13	Minivan-Standard	$152.99
14	Van-Passenger Van (12 passengers)	$161.00
15	Van-Cargo Van	$19.95
16	Pick Up-Mid Size	$69.95
17	Pick Up-Full Size	$105.99
18	Motorcycle-Touring	$19.95
19	Motorcycle-Cruiser	$199.99
20	Motorcycle-Scooter	$79.95

6. Application Physical & Software Technical Architectures

Application Physical Architecture Overview

- During the design meetings with the application architects and full-stack developers a decision was made on the ***physical application architecture*** for the **EZRental POS** application.
- After a thorough review of both the ***business requirements*** and ***technical requirements*** by the project team, theresultant decisions on architecture (s) were based on the following:

- **Rental Agency Employees:**
  - The system in our agencies used by the customer service representatives or front-line workers, must be able to quickly respond and execute the necessary requests such as

    - **POS Customer Management (Retail Customer & Corporate Customer) features** such as ***Customer Search & Print, New Customer Registration***, ***Customer Update***, ***Customer Deletion***, & ***Customer Listing functionalities***
    - **POS Vehicle Reservation, Rental & Return Management Feature** such as ***Vehicle Reservations, Vehicle Rental & Vehicle Return functionalities.***
    - **POS Vehicle Inventory Management Feature** allows inventory personnel and employees to bulk-manage vehicles such as **Cars**, **SUVs, Mini-Vans**, ***Cargo Vans,*** and other vehicles to be ***searched***, ***added***, ***updated***, ***deleted***, ***printed***, ***listed*** etc.
    - **POS Credit Card Management Feature** such as ***Credit Card Search & Print, New Credit Card Registration***, ***Credit Card Update***, ***Credit Card Deletion***, & ***Credit Card Listing functionalities.***

  - customer reservations, rentals, returns, customer management etc., therefore fast response and performance is required to quickly service a customer and minimize the wait. This is more important in Airports and other high-traffic locations.
  - We also want to provide our customer service agents with a rich user-interface experience.
  - The system in the agencies is also used by other back-end personnel such as vehicle inventory managers and administrators, service personnel, vehicle transport drivers, etc. Therefore, the system needs to also perform we

- **Corporate Offices:**
  - The corporate offices are where our business operations are managed by our business employees & employees at the rental agencies via the **INTRANET Web Portal**.
  - These features include:

    - **Intranet Web Enterprise Resource Planning Systems (ERP) Portal Feature** such as providing access to **Enterprise Resource Planning Systems (ERP) Applications** such as: ***Customer Credit Card Management System***, ***Vehicle Inventory Management System***, ***Customer Relationship Management (CRM)***, ***Human Resource Management System***, & ***Finance & Operations System, Marketing System, Customer & Field Service System etc.***
    - **Web EZRental Point-of-Sales Corporate Management System** which allows employees to ***manage*** & ***execute*** **Point-of-Sales (POS)** transactions via the **Intranet Web Poral** such as: ***Search Customer Profile Information***, ***Customer Account Management***, ***Customer Registration***, ***Customer Update***, ***Customer Delete***, & ***Customer Listing functionalities, Manage & Make Reservations of a Vehicle***, ***Manage an existing Rental***, ***etc.***

  - The system should also perform well, but the performance requirements are not as stringent as our rental agencies for which the Corporate Web Intranet meets these requirements.

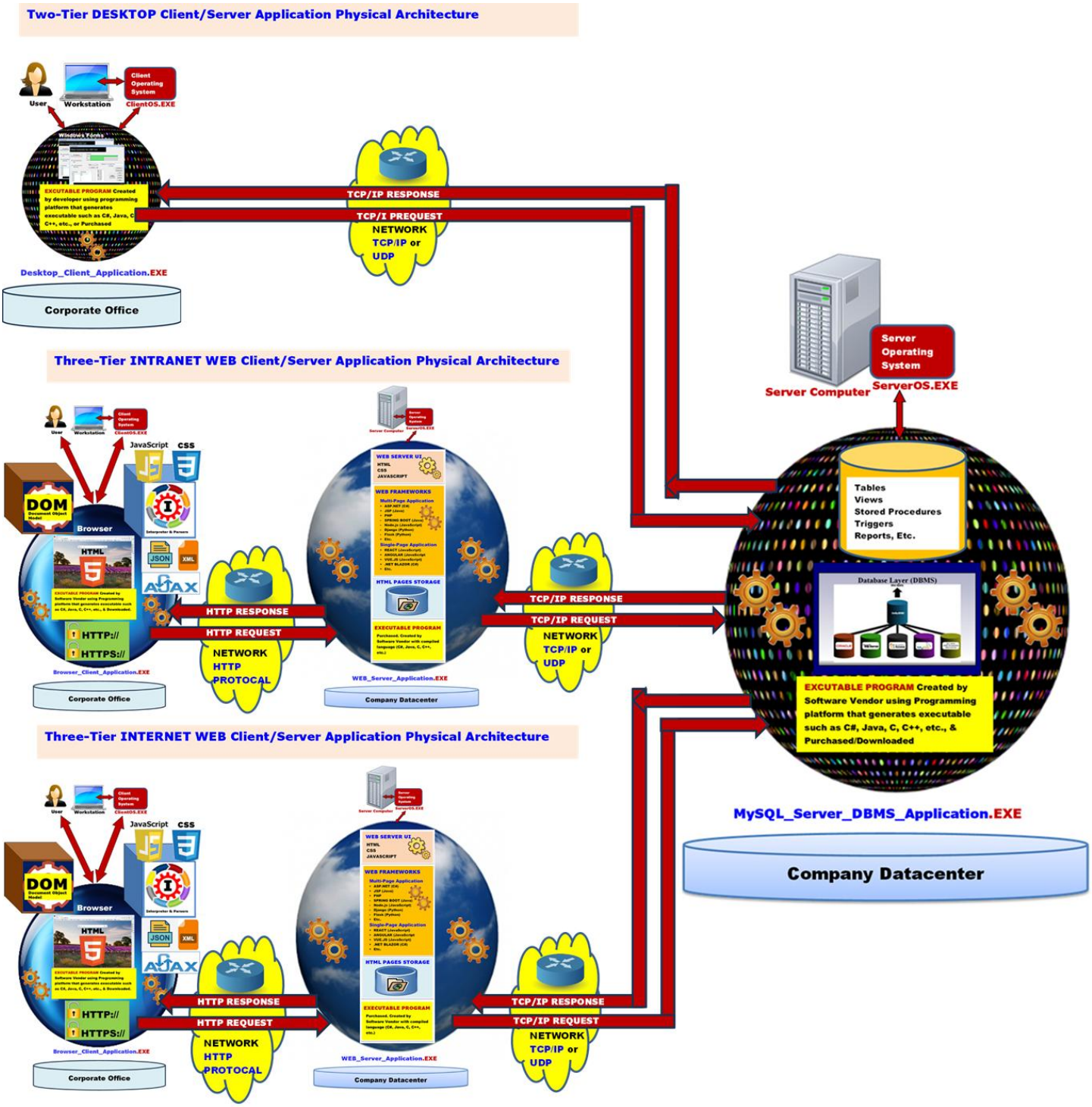
- **Customer self-service Web Portal:**
  - Customers who wish to make reservations and manage their reservations and rentals online via the internet, should be able to do so from anywhere in the world via an **INTERNET Web Portal**.
  - Features include:
    - **Web Customer Facing EZRental Point-of-Sales System** which allows customers to ***manage*** & ***execute*** **Point-of-Sales (POS)** transactions online such as **reservations** & **Profile** online via a **BROWSER** using an **Internet Web Poral**. This includes functionality such as ***Search Customer Profile Information***, ***Customer Account Management***, ***Customer Registration***, ***Customer Update***, ***Customer Delete***, & ***Customer Listing functionalities, Manage & Make Reservations of a Vehicle***, ***Manage an existing Rental***, ***etc.***
  - The system should also have good user-experience and performance.

- Based on the above requirements and ideation, the derived target applications architecture and components are as follows:

- **Rental Agency Two-Tiered Windows Desktop Client/Server Application –** Front-line workers such as customer service desk in store branches, airports etc., in addition to other support personnel such as service centers employees, inventory etc., are to use this Windows-based client application for speed and performance.
- **Corporate Office Three-Tiered Web-based Client/Server –** This Web Application named EZRentalCorp.com, targeted for corporate business users in the corporate offices to manage the day-to-day business activities of our business and office workers personnel via a Browser Application.
- **Customer Internet Three-Tiered Web-based Client/Server –** This Web Application named EZRental.com, targeted for customers who will reserve vehicles online via a Browser Application.
- **Database Tier supporting all Three Applications (Rental Agency, Corporate Office & Customer Internet) –** Using **MS SQL Server for CST4708 class**. All the front-end applications (***Two-Tier Window for agencies***, ***Three-tiered Web for Corporate Offices,*** and ***Three-Tier Web for Customers Internet application***) will **SHARE** the same **DATABASE TIER**. More information on the database scope will be provided in sections to follow.

❖ **YOUR GOAL IN THIS PROJECT #1 IS TO DESIGN & IMPLEMENT THE DBMS SERVER APPLICATION FOR THE THREE APPLICATIONS which includes the Two-Tiered Windows Desktop Client/Server Application for Rental Agencies, the Three-Tiered Web-based Client/Server for Corporate Offices & Three-Tiered Web-based Client/Server for Customer Internet Application!**

- Below is a pictorial diagram of this multi-component client/server architecture. Note that both the **Windows Desktop Client Application**, the **Corporate Office Browser Web Client Applications**, and the **Customer Internet Browser Web Client Application** are all sharing the same **Microsoft SQL Server DBMS Server Application**:


Two-Tier DESKTOP Client/Server Application Physical Architecture
User
Workstation
Client Operating System
ClientOS.EXE
EXCUTABLE PROGRAM Created by developer using programming platform that generates executable such as C#, Java, C, C++, etc., or Purchased
Desktop_Client_Application.EXE
Corporate Office
TCP/IP RESPONSE
TCP/I PREQUEST
NETWORK
TCP/IP or
UDP
Server Computer
Server Operating System
ServerOS.EXE
Three-Tier INTRANET WEB Client/Server Application Physical Architecture
JavaScript
CSS
DOM
Browser
HTML
JSON
XML
AJAX
HTTP://
HTTPS://
Browser_Client_Application.EXE
Corporate Office
HTTP RESPONSE
HTTP REQUEST
NETWORK
HTTP
PROTOCAL
WEB SERVER UI
HTML
CSS
JAVASCRIPT
WEB FRAMEWORKS
Multi-Page Application
Single-Page Application
HTML PAGES STORAGE
EXECUTABLE PROGRAM
WEB_Server_Application.EXE
Company Datacenter
TCP/IP RESPONSE
TCP/IP REQUEST
NETWORK
TCP/IP or
UDP
Tables
Views
Stored Procedures
Triggers
Reports, Etc.
Database Layer (DBMS)
EXCUTABLE PROGRAM Created by Software Vendor using Programming platform that generates executable such as C#, Java, C, C++, etc., & Purchased/Downloaded
MySQL_Server_DBMS_Application.EXE
Company Datacenter
Three-Tier INTERNET WEB Client/Server Application Physical Architecture
JavaScript
CSS
DOM
Browser
HTML
JSON
XML
AJAX
HTTP://
HTTPS://
Browser_Client_Application.EXE
Corporate Office
HTTP RESPONSE
HTTP REQUEST
NETWORK
HTTP
PROTOCAL
WEB SERVER UI
HTML
CSS
JAVASCRIPT
WEB FRAMEWORKS
Multi-Page Application
Single-Page Application
HTML PAGES STORAGE
EXECUTABLE PROGRAM
WEB_Server_Application.EXE
Company Datacenter
TCP/IP RESPONSE
TCP/IP REQUEST
NETWORK
TCP/IP or
UDP

Physical Architecture Hardware & Software Inventory

- The list of hardware and software required for purchasing/downloading is shown in the table below.

Architecture Component	Hardware Purchase & Inventory	Software Purchase & Inventory
Rental Agency Windows Client/Server Infrastructure	**User Desktop/Laptop Computer:** ▪ As needed, purchase/upgrade Memory, Processor, Hard disk etc., only if PCs need to be upgraded to support the Windows Client application. **Network Hardware:** ▪ Required Switches, Routers & other network peripherals required to support the networking requirements of the application. **Office Desktop/Laptop Installation:** ▪ Assemble team to install the Windows Client Application to user's computers or use Computer Management Tool to package and deploy the applications to the user's computers.	**User Desktop/Laptop Operating System:** ▪ Target OS – Windows 10, MAC Os etc. **Application Development & Framework:** ▪ Purchase/download required Application Development Tools such as Visual Studio, Eclipse, NetBeans etc., to develop the Windows Client Application. ▪ Download/Purchase any required framework for developing the Windows Client Application.
Corporate Office Intranet Web Client/Server Infrastructure	**Physical Server:** ▪ 1 Physical Server Computer with required specifications. **Network Hardware:** ▪ Required Switches, Routers & other network peripherals. **Datacenter Hardware Installation:** ▪ Required racks, cooling, electrical and other hardware to support installation of physical servers, etc.	**Server Operating System:** ▪ Target OS – Windows Server, Linux, Unix, etc. **Web Server Application:** ▪ Purchase/download target Web Server Application such as MSFT IIS, Apache, other. **Web Development Framework:** ▪ Purchase/download required Web Development Tools ▪ Purchase/download required web development framework such as React, Angular, ASP.NET etc.
Customer Facing Internet Web Client/Server Infrastructure	**Physical Server:** ▪ 1 Physical Server Computer with required specifications. **Network Hardware:** ▪ Required Switches, Routers & other network peripherals. **Datacenter Hardware Installation:** ▪ Required racks, cooling, electrical and other hardware to support installation of physical servers, etc.	**Server Operating System:** ▪ Target OS – Windows Server, Linux, Unix, etc. **Web Server Application:** ▪ Purchase/download target Web Server Application such as MSFT IIS, Apache, other. **Web Development Framework:** ▪ Purchase/download required Web Development Tools ▪ Purchase/download required web development framework such as React, Angular, ASP.NET etc.

Shared Database Management System Infrastructure for Agency Two-Tier Client Server, Office, and Customer Web Portals	**Physical Server:** ▪ 1 Physical Server Computer with required specifications. **Network Hardware:** ▪ Required Switches, Routers & other network peripherals. **Datacenter Hardware Installation:** ▪ Required racks, cooling, electrical and other hardware to support installation of physical servers, etc.	**Server Operating System:** ▪ Target OS – Windows Server, Linux, Unix, etc. **Database Management System Application:** ▪ Purchase/download target DBMS Server Application, which in this case is either **Microsoft SQL Server** or **Oracle 18c Express Edition**. ▪ Purchase/download the **DBMS Dev & Admin tools** such as **MS SQL Server Management Studio** for **Microsoft DBMS**, or **Oracle SQL Developer** for **Oracle DBMS** etc.

Application Software Architecture Requirements

Application Software Architecture Overview

- During the design meetings with the application architects and full-stack developers, from the ***physical application architecture*** decisions, a decision was also made on the ***software application architecture*** to be used to ***design*** and ***program*** the **EZRental POS** application for each of the ***physical architectures***:

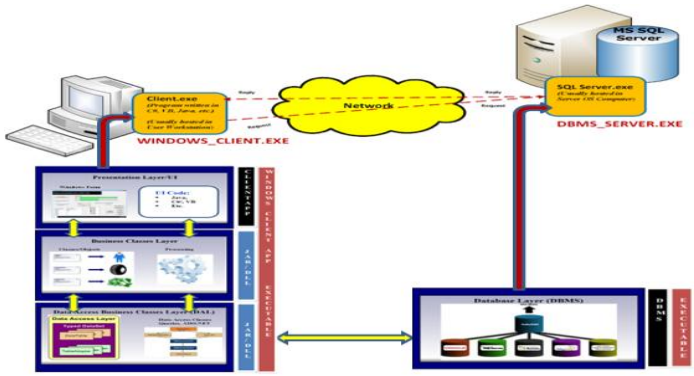
  - **Rental Agency Two-Tiered Windows Desktop Client/Server Application**.
  - **Corporate Office Two-Tiered Web-based Client/Server**.
  - **Customer Facing Internet Two-Tiered Web-based Client/Server**.
  - All sharing the **Database Tier supporting all Two Applications (Rental Agency, Corporate Office & Customer Internet)**.

Rental Agency Two-Tiered Windows Desktop Client/Server Application

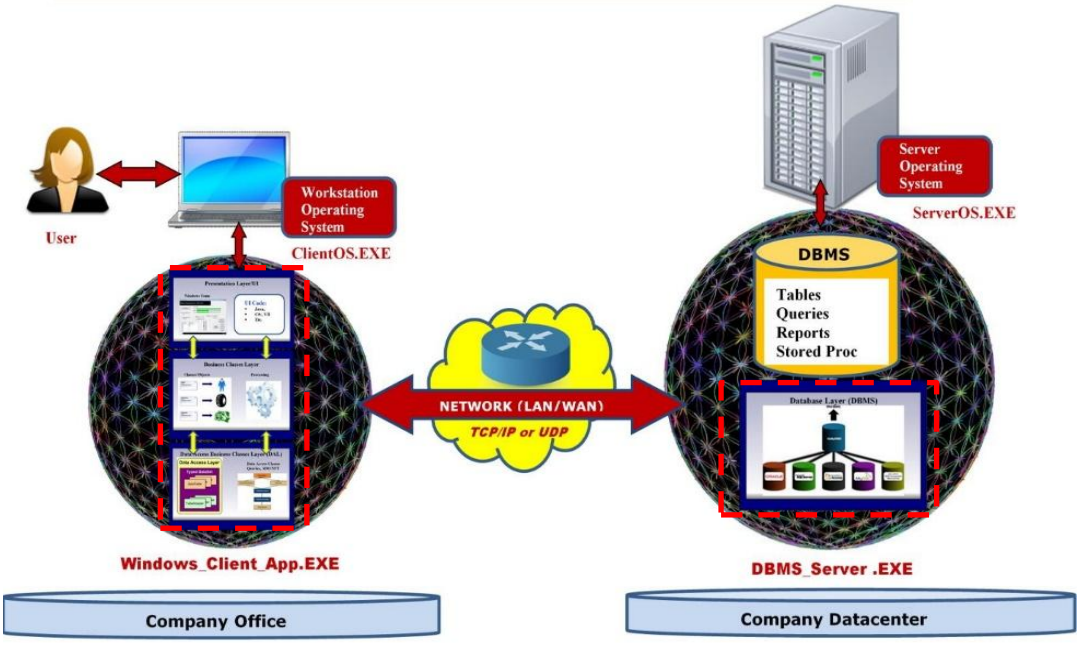
- For the Rental Agencies **Two-Tiered Windows Desktop Client/Server Application** we will use the scalable **Four-Layer Windows Client/Server Software Programming Architecture**:

- The layers will be programed as shown below as a **Windows Desktop FAT CLIENT Application** requesting database services from the **Database Management System Application**:

  - High-level view of software architecture embedded within the physical architecture:



Two-Tier Client/Server Application Programming Methodology



7.Application Development Features and Functionalities (Agile Backlog)

Features & Functionality Overview

During analysis and meetings, it was decided that the application is to deliver the following 11 features & functionality. These 11 features together make up the AGILE BACKLOG.

Feature #	Feature Description
FEATURE #1A	FEATURE #1A – EZRental Rental Agency Point-of-Sales (POS) System CUSTOMER SERVICE – DESKTOP APPLICATION CUSTOMER MANAGEMENT SYSTEM: ▪ **DESKTOP APPLICATION POINT-OF SALES SYSTEM** – ***WINDOWS UI FORM(S) FRONT-END APPLICATION & OOP PROGRAMMING*** features used by ***customer service representative employees*** via the ***Point-of-Sales computer*** machine in the ***Rental Agencies*** to service customer's **CUSTOMER MANAGEMENT** requests or transactions. ▪ The following are features and functionality that are required for this application feature: o POS Customer Management Feature: **POS Customer Management (Retail Customer & Corporate Customer) features** such as ***Customer Search & Print, New Customer Registration***, ***Customer Update***, ***Customer Deletion***, & ***Customer Listing functionalities***. o **Note that each transaction is saved to database immediately after execution:** - **Feature UI Form Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to support this feature. ▪ This feature is designed only to be used by customer service agents and other employees using the **DESKTOP APPLICATION Two-Tiered Client/Server Application** in the **Rental Agencies**.
FEATURE #1B	FEATURE #1B – EZRental Rental Agency Point-of-Sales (POS) Customer Management System Back-end Database Design & Implementation to support this feature: ▪ **DATABASE SERVER BACK-END SYSTEM –** ***BACK-END DATABASE DESIGN & FEATURES*** **(Create the Tables & Relationships, pre-populated tables, stored procedures, views, indexes etc.,) to support this feature.**

Feature #	Feature Description
FEATURE #2A	FEATURE #2A – EZRental Rental Agency Point-of-Sales (POS) System CUSTOMER SERVICE VEHICLE RESERVATION, RENTAL & RETURN FEATURE MANAGEMENT DESKTOP APPLICATION: ▪ **DESKTOP APPLICATION POINT-OF SALES SYSTEM** – ***WINDOWS UI FORM(S) FRONT-END APPLICATION & OOP PROGRAMMING*** features used by ***customer service representative employees*** via the ***Point-of-Sales computer*** machine in the ***Rental Agencies*** to service customer's **CUSTOMER VEHICLE RESERVATION, RENTAL & RETURN MANAGEMENT,** or transactions. ▪ The following are features and functionality are required for this application feature: o POS Vehicle Reservation, Rental & Return Management Feature: **POS Customer Vehicle Reservation, Rental & Return Management (Retail Customer & Corporate Customer) features** such as ***Vehicle Reservations, Vehicle Rental & Vehicle Return functionalities:*** - **Feature UI Form Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to support this feature. ▪ This feature is designed only to be used by customer service agents and other employees using the **DESKTOP Two-Tiered Client/Server Application** in the **Rental Agencies**.
FEATURE #2B	FEATURE #2B – EZRental Rental Agency Point-of-Sales (POS) Customer Vehicle Reservation, Rental & Return Management System Back-end Database Design & Implementation to support this feature: ▪ **DATABASE SERVER BACK-END SYSTEM –** ***BACK-END DATABASE DESIGN & FEATURES*** **(Create the Tables & Relationships, pre-populated tables, stored procedures, views, indexes etc.,) to support this feature.**

Feature #	Feature Description
FEATURE #3A	FEATURE #3A – EZRental Internal Back-Office Agency BACK-OFFICE VEHICLE INVENTORY MANAGEMENT SYSTEM DESKTOP APPLICATION (NOT A CUSTOMER FACING APPLICATION): ▪ **DESKTOP AAPPLICATION POINT-OF SALES SYSTEM** – ***WINDOWS UI FORM(S) FRONT-END APPLICATION & OOP PROGRAMMING*** features used by **Back-Office Inventory Team employees** via a **computer machine** in the ***Rental Agencies*** to service inventory needs for **VEHICLE INVENTORY MANAGEMENT,** or transactions. ▪ This is a unique Back-end system meant for inventory team employees to perform bulk IN-MEMORY inventory processing or management tasks on vehicles such as: ***adding*** vehicles to the system, ***searching*** for vehicles, ***updating*** vehicles, ***deleting*** vehicles, etc. The idea is that the employee can perform all these features on their computer **in-memory** repeatedly for several vehicles in one session saving to database after each transaction but managed in-memory using a collection or other data structure to manage it locally. When user is done with all inventory transactions, all transactions have been saved to database but, is still locally in the collection or other data structure and can be updated as needed. By keeping it locally in memory, the operations are faster. ▪ The following are features and functionality are required for this application feature: o POS Vehicle Inventory Management Feature: **POS Vehicle Inventory Management features** allows inventory personnel and employees to bulk-manage vehicles such as Cars, SUVs, Mini-Vans, ***Cargo Vans,*** and other vehicles to be ***searched***, ***added***, ***updated***, ***deleted***, ***printed***, ***listed*** etc. - **Feature UI Form Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to in the to support this feature. ▪ This back-office features are not designed to be used by customers and not available via the Web and implemented using the **DEKTOP Two-Tiered Client/Server Application** in the **Rental Agencies**.
FEATURE #3B	FEATURE #3B – EZRental Rental Agency Vehicle Inventory Management System Back-end Database Design & Implementation to support this feature:

	▪ **DATABASE SERVER BACK-END SYSTEM – *BACK-END DATABASE DESIGN & FEATURES* (Create the Tables & Relationships, pre-populated tables, stored procedures, views, indexes etc.,) to support this feature.**

Feature #	Feature Description
FEATURE #4A	FEATURE #4A – EZRental Rental Agency Point-of-Sales (POS) BACK-OFFICE DESKTOP APPLICATION CREDIT CARD MANAGEMENT SYSTEM: ▪ **DESKTOP APPLICATION POINT-OF SALES SYSTEM** – ***WINDOWS UI FORM(S) FRONT-END APPLICATION & OOP PROGRAMMING*** features is a **back-end system** used by ***customer service representative & other employees*** via the ***Point-of-Sales computer*** machine in the ***Rental Agencies*** to service customer's **CREDIT CARD MANAGEMENT,** or transactions required when servicing customers. ▪ The following are features and functionalities required for this application with features such as: o POS Credit Card Management Feature: **POS Customer Credit Card Management features** such as ***Credit Card Search & Print, New Credit Card Registration***, ***Credit Card Update***, ***Credit Card Deletion***, & ***Credit Card Listing functionalities:*** - **Feature UI Form Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to support this feature. ▪ This feature is designed only to be used by customer service agents and other employees using the **DESKTOP Two-Tiered Client/Server Application** in the **Rental Agencies**.
FEATURE #4B	FEATURE #4B – EZRental Rental Agency Point-of-Sales (POS) Credit Card Management System Back-end Database Design & Implementation to support this feature:

	▪ **DATABASE SERVER BACK-END SYSTEM – *BACK-END DATABASE DESIGN & FEATURES* (Create the Tables & Relationships, pre-populated tables, stored procedures, views, indexes etc.,) to support this feature.**

Feature #	Feature Description
FEATURE #5A	FEATURE #5A – EZRental Rental Agency Point-of-Sales (POS) System BACK-OFFICE EMPLOYEE & CUSTOMER USER-ACCOUNT MANAGEMENT SYSTEM FOR DESKTOP APPLICATION: ▪ **DESKTOP APPLICATION POINT-OF SALES SYSTEM** – ***WINDOWS UI FORM(S) BACK-OFFICE APPLICATION & OOP PROGRAMMING*** **User-Accounts Management Features** used by ***customer service representative & IT Administrator employees*** via the ***Point-of-Sales computer*** machine in the ***Rental Agencies*** to service customer's **CUSTOMER & EMPLOYEE USER ACCOUNT MANAGEMENT** requests or transactions. ▪ Employee User Accounts – These are the user accounts used by ***IT Administrators***, ***Customer Service Employees***, ***back-office employees***, and ***any employee who qualifies*** for access to the system. ▪ Customer User Accounts – These are the user accounts used by ***IT Administrators***, ***Customer Service Employees***, ***back-office employees*** and any ***employee*** who has access to the system to ***manage*** the Customer User Accounts for login into the Customer Web Portal. ▪ The following are features and functionality that are required for this application feature: o POS User Account (Employee & Customer) Management Feature: **POS User Account Management (Employee & Customer) features** such as: - **Employee User Account Feature 5A-1 –** Allows ***IT Administrators***, ***Customer Service Employees***, ***back-office employees***, and ***any employee who qualifies*** to ***manage*** employee user accounts that allow employees to login into the POS System. And perform the following tasks: ***Employee User Account Search by username, New Employee User Account Registration***, ***Employee User Account Update by username***, ***Employee User Account Deletion by username***, & ***Employee User Account Listing functionalities***. IMPORTANT! Note that the password is ***NEVER DISPLAYED or LISTED***, only the username! - **Customer User Account Feature 5A-2 –** Allows ***IT Administrators***, ***Customer Service Employees***, ***back-office employees***, and ***any employee who qualifies*** to ***manage*** customer user accounts that allow customers to login into the Customer Web Portal System. And perform the following tasks: ***Customer User Account Search by username, New Customer User Account Registration***, ***Customer User Account Update by username***, ***Customer User Account Deletion deletion by username***, & ***Customer User Account Listing functionalities***.

	IMPORTANT! Note that the password is ***NEVER DISPLAYED or LISTED***, only the username! o ***Note that each transaction is saved to database immediately after execution:*** - **Feature UI Form Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to support this feature. ▪ This feature is designed only to be used by ***IT Administrations*** and other employees who qualify to use this system to manage both employees & customers user accounts using the **DESKTOP Two-Tiered Client/Server Application** in the **Rental Agencies**.

Feature #	Feature Description
FEATURE #6A	FEATURE #6A – EZRental Rental Agency Point-of-Sales (POS) System EMPLOYEES BACK-OFFICE SECURITY LOGIN AUTHENTICATION SYSTEM FOR DESKTOP APPLICATION: ▪ Proper ***security and authentication*** must be implemented to make sure only authorized employees can access the Point-0f- Sales & Back-End Management systems when they login into the DESKTOP Two-Tiered Client/Server Application & Web Three-Tiered Corporate Client/Server Application. ▪ **WINDOWS CLIENT POINT-OF SALES SYSTEM** – ***WINDOWS UI FORM(S) BACK-OFFICE APPLICATION & OOP PROGRAMMING*** Login Authentication features used by ***customer service representative & IT Administrator employees*** via the ***Point-of-Sales computer*** machine in the ***Rental Agencies*** to service employee **LOGIN AUTHENTICATION SYSTEM**. ▪ The following are features and functionality that are required for this application such as: o POS Employee Back-Office Security Login Authentication System: **POS Login Authentication Access for Employees features** such as: - **Employee Authentication Feature 6A-1 –** To have access to the application, an employee (Customer Service Reps, Back-office employee etc.) must provide a username & password. This feature is required to be ***designed*** & ***programmed*** into the application. - **Employee Authentication Feature 6A-2 –** ***Design & programming*** of required User-Interface Forms & GUI Controls to support the Authentication System feature!

	○ Programming includes: - **Feature UI Form Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to support this feature. ▪ This feature is designed only to be used by all employees wishing access to the **DESKTOP Two-Tiered Client/Server Application POS** System in the **Rental Agencies**.

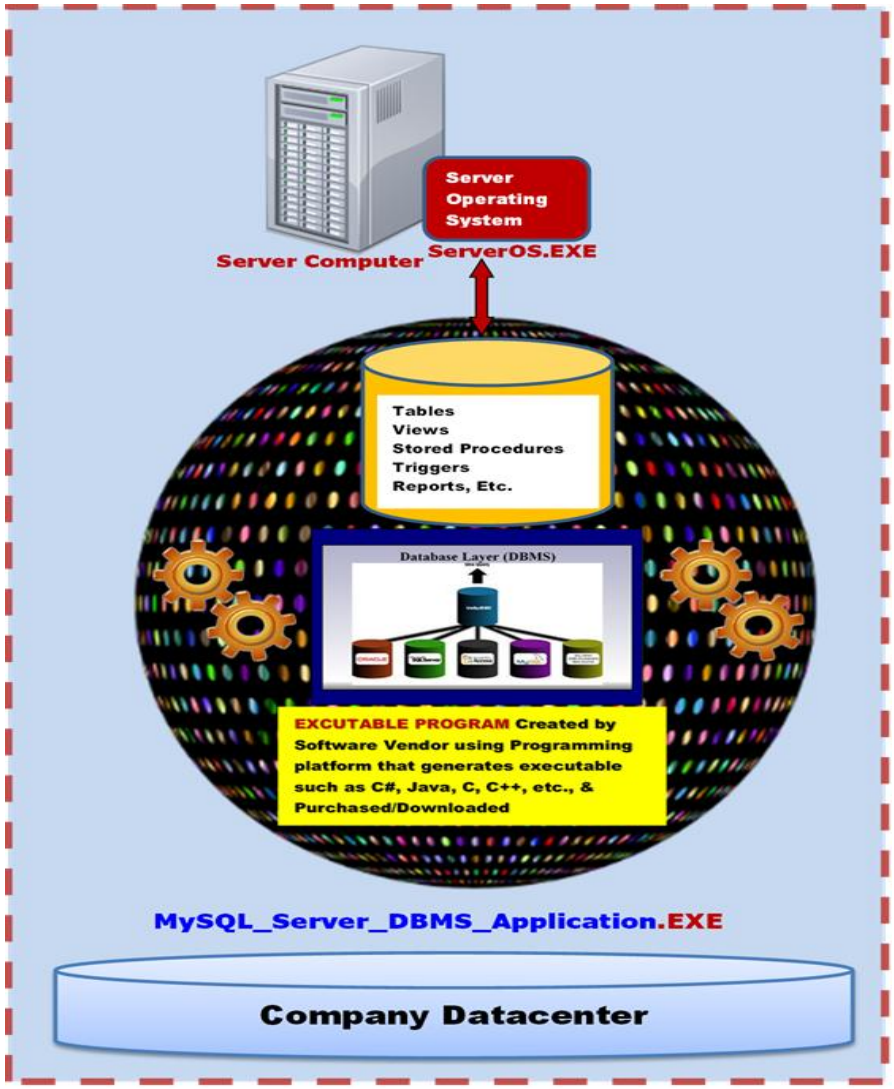
Feature #	**Feature Description**
FEATURE #7A	FEATURE #7A – **EZRental** CORPORATE OFFICES & RENTAL AGENCIES CUSTOMER SERVICE AGENTS VIA THE INTRANET CORPORATE POINT-OF-SALES (POS) BACK-OFFICE CREDIT CARD MANAGEMENT SYSTEM WEB APPLICATION: ▪ This INTRANET (NOT THE PUBLIC INTERNET) Web Portal **EZRentaCorp.com**, is a Web-based Three-Tiered Client/Server physical & Software Layered Development Architecture used by CORPORATE EMPLOYEES & RENTAL AGENCIES EMPLOYEES to ***execute*** Enterprise Resource Planning Systems (ERP) Applications & other Corporate Applications online intranet via a WEB BROWSER. ▪ **WEB APPLICATION CREDIT CARD MANAGEMENT SYSTEM ERP & CORPORATE APPLICATION SYSTEM** – ***WEB UI FORM/WEB PAGES FRONT-END & OOP PROGRAMMING BACKEND*** Enterprise Resource Planning Systems (ERP) Credit Card Management System Application used by ***Corporate Customer Service & Rental Agencies Customer Service Employees to perform Credit Card Management*** processing via the CORPORATE INTRANET ***PORTAL and NOT*** **the Desktop Two-Tiered Client/Server Application** ***of the Rental Agencies***. ▪ The following are features and functionality that are required for this Credit Card Management System Application INTRANET WEB APPLICATION: ○ POS Credit Card Management Feature: **POS Customer Credit Card Management features** such as ***Credit Card Search & Print, New Credit Card Registration***, ***Credit Card Update***, ***Credit Card Deletion***, & ***Credit Card Listing functionalities:*** - **Feature WEB UI Requirements:** ***Design & programming*** of required ***User-Interface Forms & GUI Controls*** to support this feature. - **Feature Processing Requirements:** ***Design & programming*** of required ***Object-Oriented (OOP) Processing & Logic*** to support this feature.

	▪ This feature is designed only to be used by customer service agents in and other employees in the Corporate Offices or Rental Agencies using the **WEB Three-Tiered Client/Server Application** in the **Rental Agencies**.
FEATURE #7B	FEATURE #7B – EZRental Rental Agency Point-of-Sales (POS) Credit Card Management System Back-end Database Design & Implementation to support this feature: ▪ **DATABASE SERVER BACK-END SYSTEM – *BACK-END DATABASE DESIGN & FEATURES* (Create the Tables & Relationships, pre-populated tables, stored procedures, views, indexes etc.,) to support this feature.**

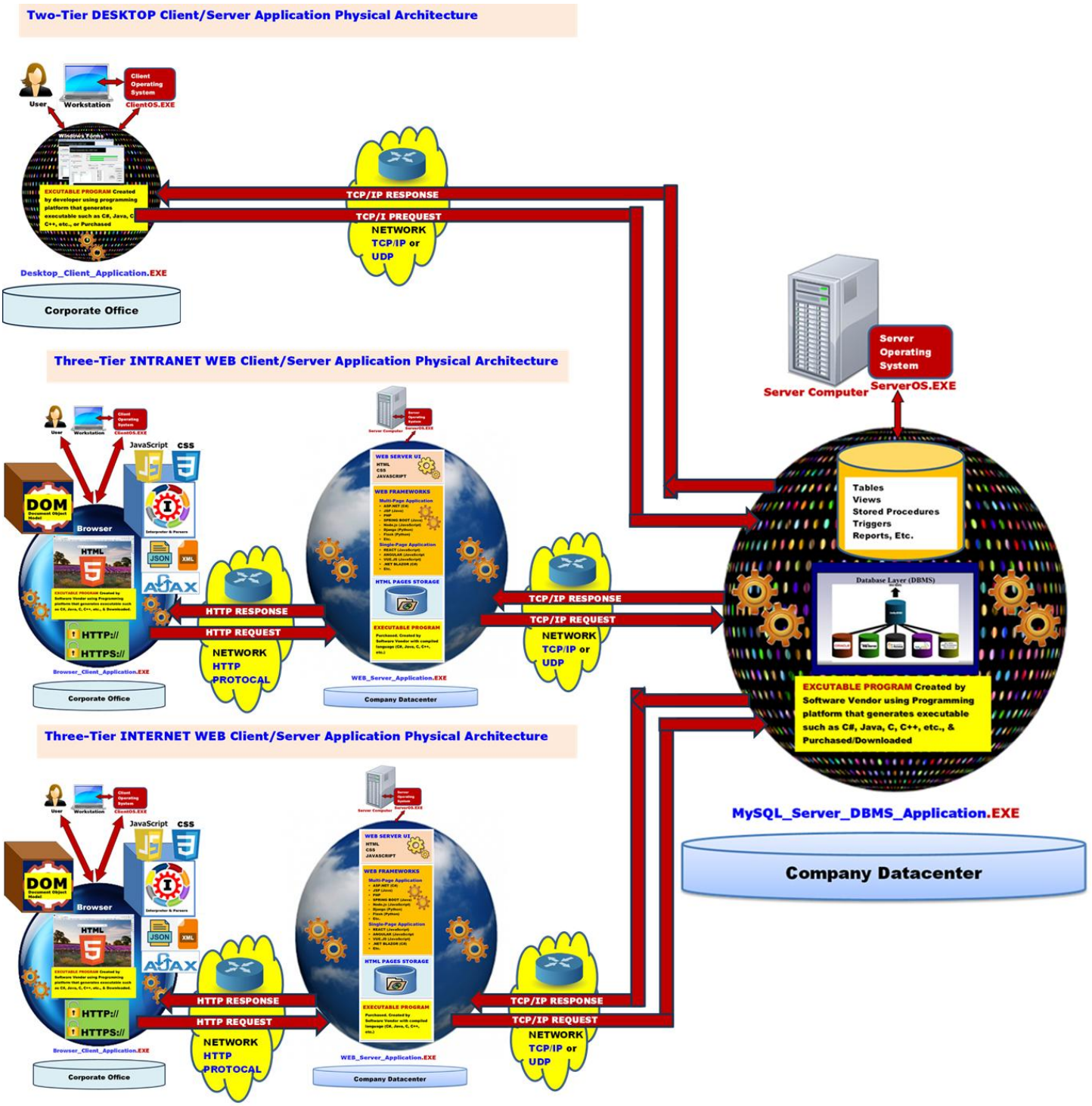
8.*Database Management System Development Environment & Physical Architecture*

Application Physical Architecture Overview

- During the design meetings with the application architects and full-stack developers a decision was made on the *physical application architecture* for the **EZRental POS** application.
- After a thorough review of both the ***business requirements*** and ***technical requirements*** by the project team, the resultant decisions on architecture (s) were based on the following:
- Below is a pictorial diagram of this multi-component client/server architecture. Note that both the **Windows Desktop Client Application**, the **Corporate Office Browser Web Client Applications**, and the **Customer Internet Browser Web Client Application** are all sharing the same **MySQL Server DBMS Server Application**:



- Another view or BIG PICTURE of the **MySQL Server DBMS Server Application** you are creating in to ***HOST*** the **DATA** for The THREE APPLICATIONS which includes the Two-Tiered Windows-Desktop Client/Server Application for the Rental Agencies, the Three-Tiered Web-based Client/Server for Corporate Offices & Three-Tiered Web-based Client/Server for Customer Internet Application:



9.Project Roles & Responsibilities

Summary of the DBMS Server Application Development Roles and Responsibilities:

The development of the EZ Auto Rental POS Management System involved a focused and structured approach to database design and implementation. The primary role in this project was that of the **Database Designer and Developer**, responsible for translating business requirements into a normalized relational database model using Microsoft SQL Server. This role included analyzing the operational needs of rental agencies, defining the data entities and their relationships, enforcing data integrity through the use of constraints, and ensuring the database was scalable and optimized for performance.

Consultant #6 **Ahmad Fatah**	**Database Administrator**	▪ The DB Admin, install the DBMS, maintain, and operate the DBMS throughoutits lifetime. ▪ Activities include but not limited to: 1. As DB Admin, you are to **1) Setup & install MySQL Server 2) Administrative tools for target DBMS**. 2. Also, as DB Admin, you are to **3) Operate & Maintain the DBMS**.

Consultant #2, 3, 4 & 5 **Ahmad Fatah**	**Database Developers**	▪ This role uses the Normalized Logical Model created by consultant #2 to create the Data Dictionary, Physical Schema Diagram, and Implement the Database Application forthe Auto Rental System. ▪ Activities include but not limited to: 1. Use the ***Normalized Logical Model*** created by consultant #2 to do the **following:** 1) **Create Data Dictionary tables for eachlogical table targeting MySQL Server .** 2) **Create Physical Schema Diagram**. 2. From these two deliverables, 1) **implement the Database Application using MySQL** for the Auto Rental System.

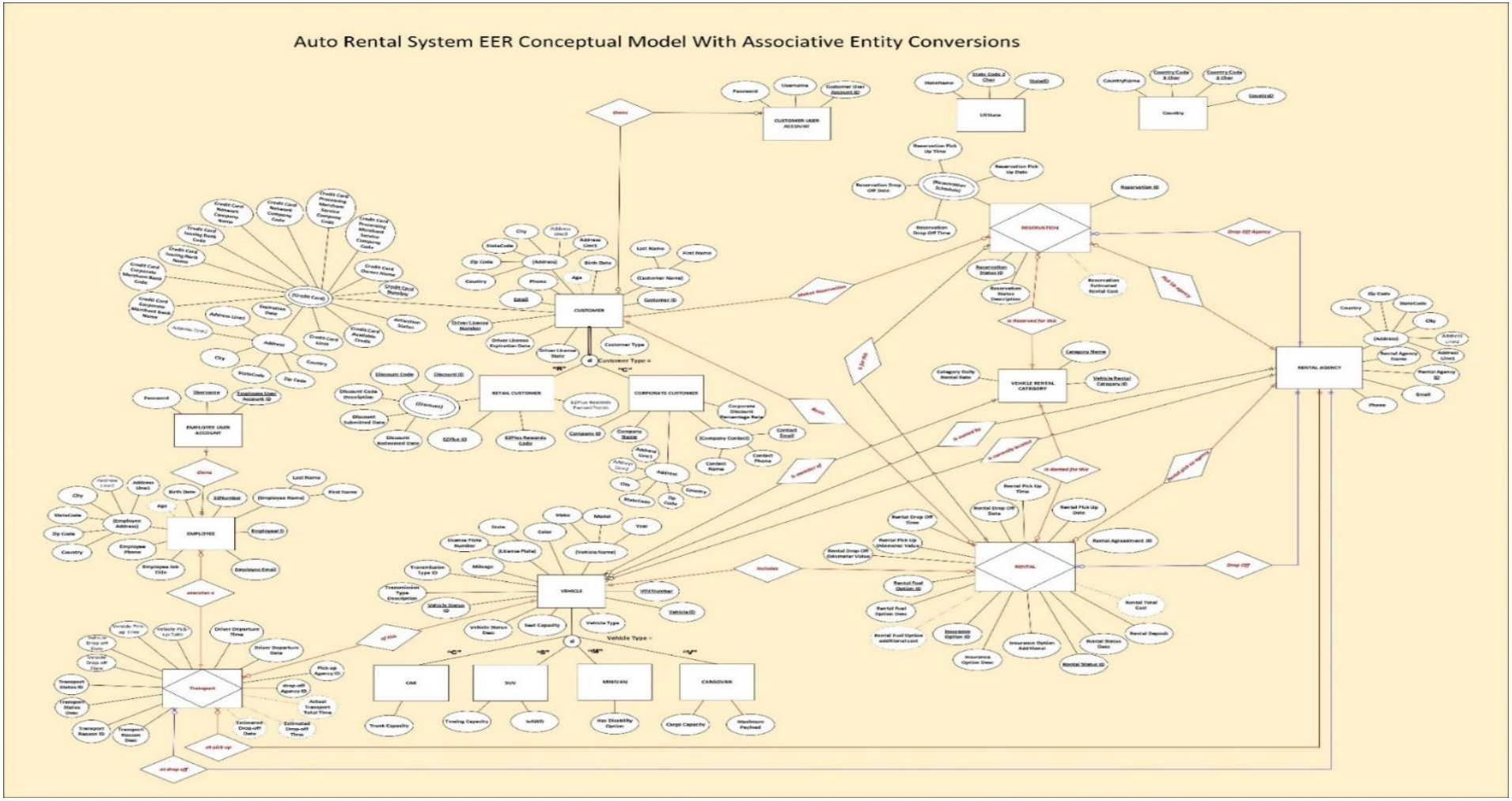
Summary of the Windows Client & Browser Client Applications Development Project Roles and Responsibilities

Consultants #8, 9, 10, 11 & 12 **Ahmad Fatah**	**Full Stack Windows Application Developers & UI/UX Client Application Developer**	▪ Object-Oriented-Programming developer to implement the **Windows Client Application** using **C# & .NET technologies** & on the database side, implement stored procedures and support the databased team as needed. ▪ Activities include but not limited to: 1. As full stack developer, **Programming & implementation of the Object-Oriented-Programming of Class/Object Model designed by consultant #7 for the Windows Client Application using C# & .NET Technologies**. 2. In addition, **Development of Database Stored Procedures,and other development requirements in the Back-end DBMS**. 3. From the technical requirements, **design a high-level GraphicalUser-Interface (GUID) wireframe**, & **implement the front-end UI Programming, features & functionality**
Consultant #14, 15, 16, 17 & 18 **Ahmad Fatah**	**Full Stack Web Application Developer & UI/UX Web Application Developer**	▪ Object-Oriented-Programming developer to implement the **Web Browser Application** using **C# & ASP.NET technologies**. ▪ Activities include but not limited to: 1. As full stack developer, **Programming & implementation of the Object-Oriented-Programming of Class/Object Model designed by consultant #7 for the Web Browser Client Application using C# & ASP.NET Technologies**. 2. From the technical requirements, **design a high-level Graphical User-Interface (GUID) wireframe**, & **implement the Webfront-end UI Programming, features & functionality in the Web Server Application**

10.

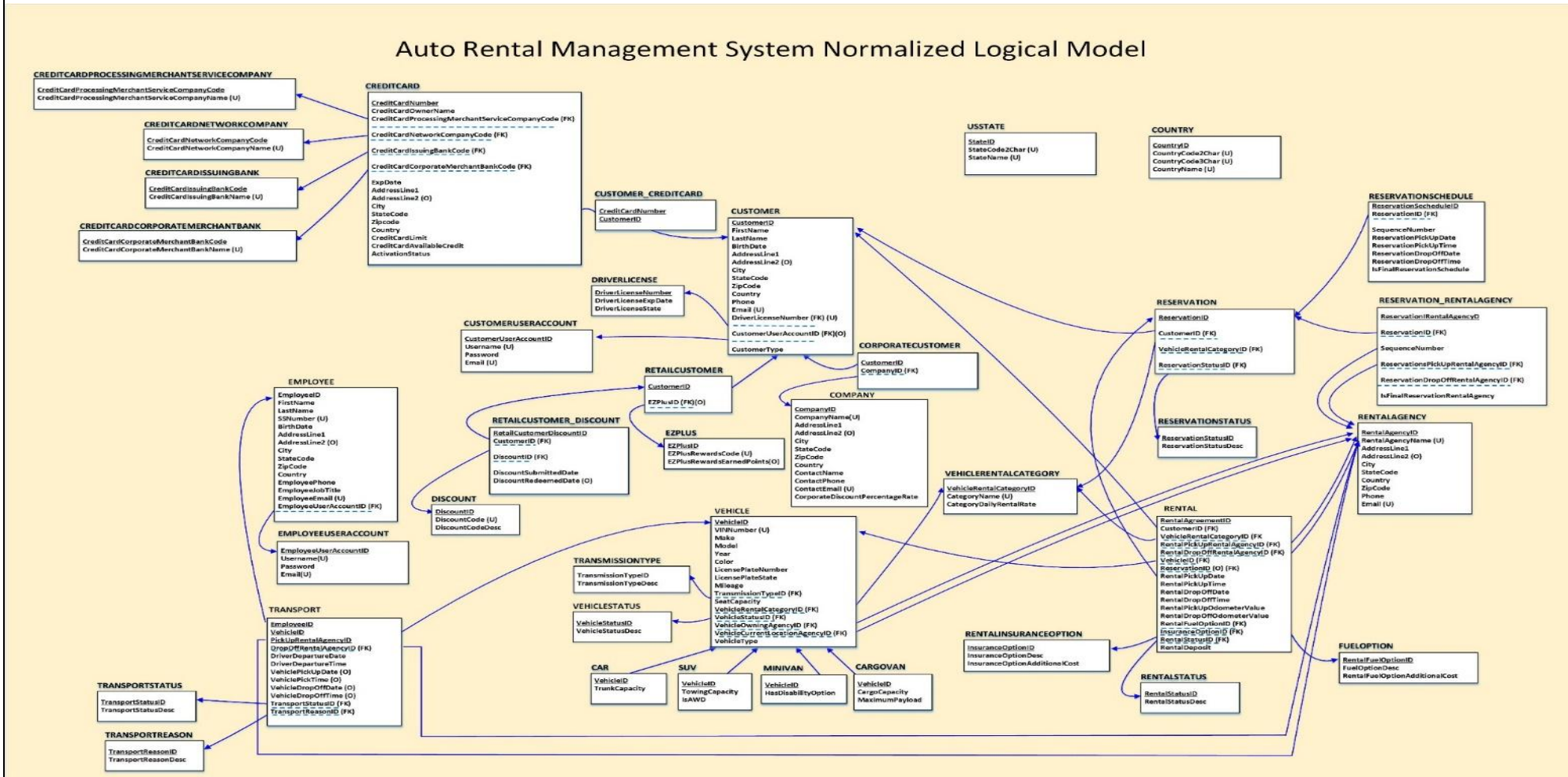
Database Design Deliverable #2 – ER/EER Conceptual Model Diagram

The EER Conceptual Model Diagram represents a critical component of the Auto Rental Management System's database design process. A Database Analyst/Architect, commissioned by project sponsor Mr. Rodriguez, developed this Enhanced Entity-Relational (EER) Conceptual Model during the Analysis Phase, drawing directly from the Application Business Requirements. This diagram serves as the foundational blueprint for the Database Management System (DBMS), illustrating the relationships and entities identified in the business requirements to create a high-level visual representation of how the application's key business data interconnects. By mapping these elements, the EER Conceptual Model Diagram provides a clear and structured overview essential for the database design, and it must be incorporated into the Design & Implementation Project Document to guide subsequent development stages.


Auto Rental System EER Conceptual Model With Associative Entity Conversions
CUSTOMER USER ACCOUNT
USState
Country
RESERVATION
CUSTOMER
RETAIL CUSTOMER
CORPORATE CUSTOMER
EMPLOYEE USER ACCOUNT
EMPLOYEE
VEHICLE
CAR
SUV
MINIVAN
CARGOVAN
VEHICLE RENTAL CATEGORY
RENTAL
RENTAL AGENCY
Transport

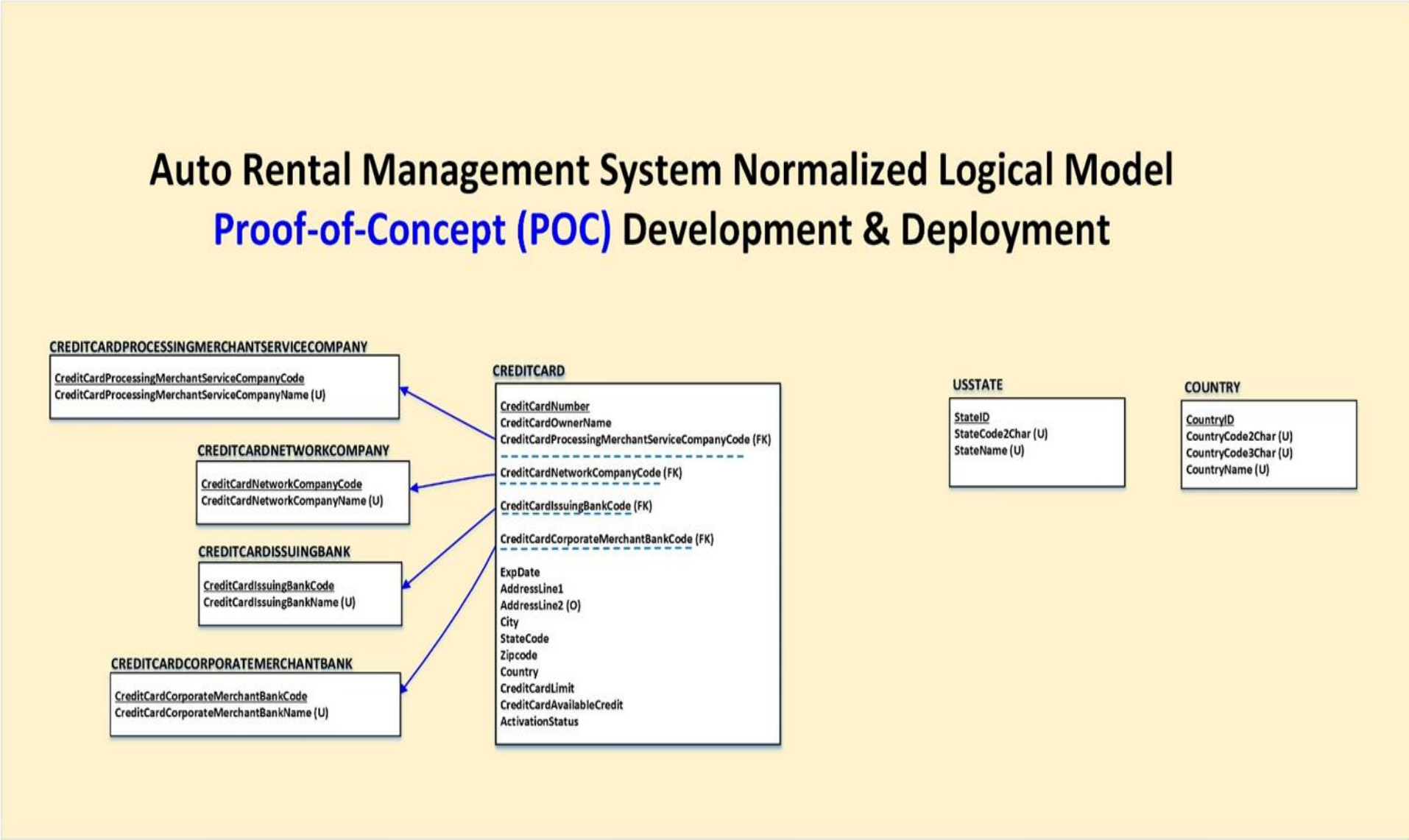
11.Database Design Deliverable #3 – Normalized Logical Model Diagram

The Normalized Logical Model Diagram represents the structured blueprint of the EZ Auto Rental database, capturing all entities, attributes, and relationships in a fully normalized form, up to the Third Normal Form (3NF). This diagram illustrates how data is logically organized, ensuring minimal redundancy, optimized data integrity, and clear relational mapping between tables. It serves as a critical reference for both development and future maintenance, providing a visual guide that aligns with the system's business rules and operational requirements.Production Normalized Logical Mode



Normalized Logical Model PROOF-OF-CONCEPT PROTOTYPE:

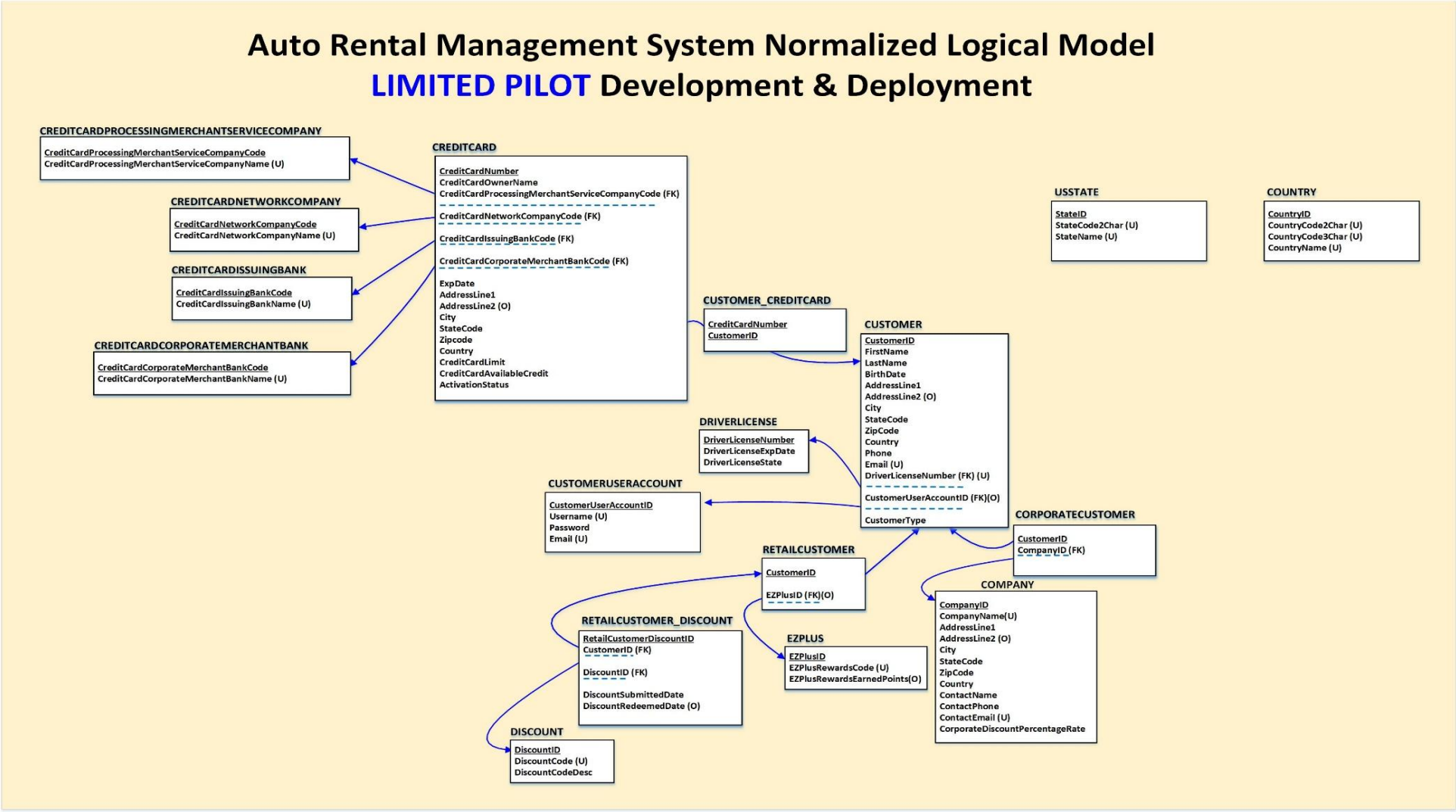
Below is the PROOF-OF-CONCEPT (POC) Normalized Logical Model that was used to create a PROTOTYPE of the application to demo to the business. This diagram only contains the 7 logical tables in scope of the POC:



Pilot normalized logical model:

Normalized Logical Model LIMITED PILOT DEPLOYMENT

- Below is the LIMITED PILOT **Normalized Logical Model** to deploy the application to a limited number of users throughout the organization as an MVP (Minimum Viable Product) or product with enough features to get feedback from users before finalizing the entire application. This diagram only contains the **17 logical tables** in scope of the PILOT:



12.Database Design Deliverable #4 -Physical Model Data Dictionary

For our EZRental-car service, we've designed a database that keeps every rental and credit card transaction secure, accurate, and seamless—ensuring what we call *data integrity*. Think of it as a system that guarantees every car booking, from a customer's reservation to their payment, stays perfectly on track, even during your busiest days. We use unique IDs—like Rental_ID for each booking and Customer_ID for every renter—to avoid mix-ups, while linking rentals to credit card details ensures no payment gets lost. Must-have fields, like Card_Number (encrypted for safety) and Rental_Date, mean nothing gets skipped, and rules like 'valid card expiration dates' or 'positive rental fees' catch errors upfront. Our tables lay this out simply, and the data dictionary explains it all—showing how fields like Cardholder_Name (text, required) or Credit_Card_Number (unique, secure) are set up to protect your business and your customers. It's all built to give you peace of mind: trustworthy data that keeps rentals rolling and payments flowing smoothly

The physical model Data Dictionary is the Second Component of the Waterfall Methodology design phase of physical model design. Here is the normalized logical model diagram.

#1:

CREDITCARD							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required ?**	**Length /Size /Format**	**Constraints**	**Description/ purpose**
1.	**CreditCardNumber**	**String**	**VARCHAR(16)**	**Yes**	**16**	**PRIMARY KEY**	***Unique Identifier*** or **Primary Key** for the CreditCard table. This **Primary Key HAS Business Meaning**.
2.	**CreditCardOwnerName**	**String**	**VARCHAR(50)**	**Yes**	**50**	**NOT NULL**	Credit Card Owner first, last and middle initial if included in name

3.	**CreditCardProcessingMerchantServiceCompanyCode (FK)**	**Number**	**TINYINT**	**Yes**	**20**	**CHECK(CreditCardProcessingMerchantServiceCompanyCode between 1 and 20) NOT NULL**	Column stores the Credit Card Processing Merchant Service Company unique code. **Primary key** to ***CreditCardProcessingMerchantServiceCompany*** Table.
4.	**CreditCardNetworkCompany Code (FK)**	**Number**	**TINYINT**	**Yes**	**20**	**CHECK(CreditCardProcessingMerchantServiceCompanyCode between 1 and 20) NOT NULL**	Stores the Credit Card Network Company unique code. **Unique code or Primary** to ***CreditCardNetworkCompany*** Table.
5.	**CreditCardIssuingBankCode (FK)**	**Number**	**TINYINT**	**Yes**	**20**	**CHECK(CreditCardIssuingBankCode between 1 and 20) NOT NULL**	Stores the Credit Card Issuing Bank unique code. **Unique identifier or primary key** to ***CreditCardIssuingBank*** Table.
6.	**CreditCardCorporate MerchantBankCode (FK)**	**Number**	**TINYINT**	**Yes**	**20**	**CHECK (CreditCardCorporate MerchantBankCode BETWEEN 1 AND 20) NOT NULL**	Stores the Credit Card Corporate Merchant Bank unique code. **Foreign Key** to ***CreditCardCorporateMerchantBank*** Table.
7.	**ExpDate**	**Date**	**DATE**	**Yes**	**MM-DD-YYYY**	**NOT NULL**	The date the credit card expires.
8.	**AddressLine1**	**String**	**VARCHAR(50)**	**Yes**	**50**	**NOT NULL**	Stores house/building number & street (part 1 of address).
9.	**AddressLine2 (O)**	**String**	**VARCHAR(50)**	**No**	**50**	**NULL**	Optional value that stores remaining part of address such as apartment number, or other address information.
10.	**City**	**String**	**VARCHAR(50)**	**Yes**	**50**	**NOT NULL**	Stores the city name

11.	**StateCode**	**Characters**	**CHAR(2)**	**Yes**	**2**	**NOT NULL**	Stores the U.S. State 2-character code. E.g., NY, NJ, CT, etc.
12.	**Zipcode**	**String**	**VARCHAR(10)**	**Yes**	**10**	**NOT NULL**	Stores US Zip Code/Postal Code. Support format: xxxxx-xxxx.
13.	**Country**	**String**	**VARCHAR(100)**	**Yes**	**100**	**NOT NULL**	Stores the name of the country.
14.	**CreditCardLimit**	**Number**	**DECIMAL(8,2)**	**Yes**	**X = 8** **Y = 2**	**NOT NULL**	The maximum amount of dollars that can be charged to the credit card. Assumes that the credit card has the full limit available. The format is **DECIMAL(X,Y),**where X = The total number of digits and Y = total number of digits to the right of the decimal point. We assume the maximum number that can be stored is 999999.99 for a maximum limit amount $999,999.99, since we don't expect a customer to have a credit limit of $1 Million dollars, we cap it at $999,999.99.
15.	**CreditCardAvailableCredit**	**Number**	**DECIMAL(8,2)**	**Yes**	**X = 8** **Y = 2**	**NOT NULL**	Stores the current remaining credit available for charging.
16.	**ActivationStatus**	**Boolean**	**BIT**	**Yes**	**1**	**NOT NULL**	Stores a Boolean value indicating **True** if credit card is active or **False** otherwise. The **MySQL SERVER DBMS** has a Data Type named **BIT** that will store **1** to represent **True** and **0** to represent **False**.

#2:

CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required?**	**Length/Size /Format**	**Constraints**	**Description/ purpose**
1.	**CreditCardProcessingMerchantServiceCompanyCode**	**Number**	**TINYINT**	**Yes**	**20**	**PRIMARY KEY CHECK(CreditCard Processing MerchantService CompanyCode between 1 and 20)**	***Unique Identifier*** or **Primary Key** for this table. This **Primary Key HAS Business Meaning**.
2.	**CreditCardProcessingMerchantServiceCompanyName (U)**	**String**	**VARCHAR (50)**	**Yes**	**50**	**UNIQUE NOT NULL**	Stores the ***unique*** name of the Credit Card Processing Merchant Service Company.

#3:

CREDITCARDNETWORKCOMPANY							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required?**	**Length/Size /Format**	**Constraints**	**Description/ purpose**
1.	**CreditCardNetworkCompanyCode**	**Number**	**TINYINT**	**Yes**	**20**	**Primary Key NOT NULL Check between 1 and 20**	***Unique Identifier*** or **Primary Key** for this table. This **Primary Key HAS Business Meaning**.
2.	**CreditCardNetworkCompanyName (U)**	**String**	**VARCHAR(50**	**Yes**	**50**	**UNIQUE NOT NULL**	Stores the ***unique*** name of the Credit Card Network Company.

#4:

CREDITCARDISSUINGBANK							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required?**	**Length/Size /Format**	**Constraints**	**Description/ purpose**
1.	**CreditCardIssuingBankCode**	**Number**	**TINYINT**	**Yes**	**20**	**PRIMARY KEY CHECK(between 1 and 20)**	***Unique Identifier*** or **Primary Key** for this table. This **Primary Key HAS Business Meaning**.
2.	**CreditCardIssuingBankName (U)**	**String**	**VARCHAR(20)**	**Yes**	**26**	**UNIQUE NOT NULL**	Stores the ***unique*** name of the Credit Card Processing Merchant Service Company.

#5:

CREDITCARDCORPORATEMERCHANTBANK							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required?**	**Length/Size /Format**	**Constraints**	**Description/ purpose**
1.	**CreditCardCorporateMerchantBankCode**	**Number**	**TINYINT**	**Yes**	**10**	**PRIMARY KET CHECK (between 1 and 20)**	***Unique Identifier*** or **Primary Key** for this table. This **Primary Key HAS Business Meaning**.
2.	**CreditCardCorporateMerchantBankName (U)**	**String**	**VARCHAR(30)**	**Yes**	**30**	**UNIQUE NOT NULL**	Stores the ***unique*** name of the Credit Card Corporate Merchant Bank.

#6:

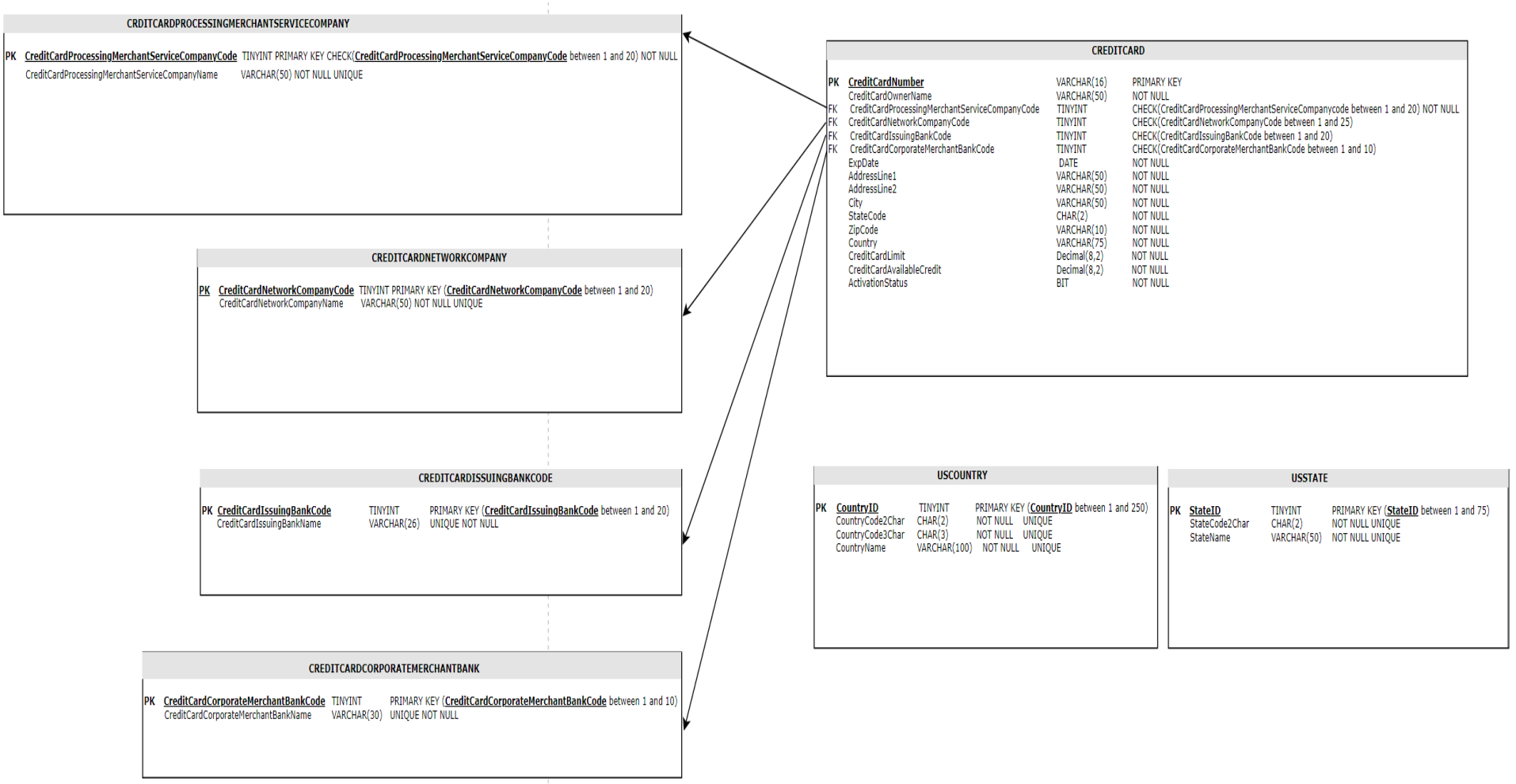
USSTATE							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required?**	**Length/Size /Format**	**Constraints**	**Description/ purpose**
1.	**StateID**	Number	TINYINT	**Yes**	**75**	**PRIMARY KEY CHECK(StateID between 1 and 75)**	*Unique Identifier* or Primary Key for this table. This Primary Key HAS **Business Meaning**.
2.	**StateCode2Char (U)**	Characters	CHAR(2)	**Yes**	**2**	**NOT NULL UNIQUE**	Stores the U.S. State 2-character code. E.g., NY, NJ, CT etc.
3.	**StateName (U)**	String	VARCHAR(50)	**Yes**	**50**	**NOT NULL UNIQUE**	Stores the unique full name of the U.S. State.

#7:

COUNTRY							
Column Num.	**Attribute/Column Name**	**Generic Data Type Name**	**MySQL SERVER Data Type Name**	**Is it Required?**	**Length/Size /Format**	**Constraints**	**Description/ purpose**
1.	**CountryID**	Number	TINYINT	Yes	**250**	**PRIMARY KEY CHECK(CountryID Between 1 and 250)**	*Unique Identifier* or Primary Key for this table. This Primary Key HAS **Business Meaning**.
2.	**CountryCode2Char (U)**	Characters	CHAR(2)	Yes	**2**	**NOT NULL UNIQUE**	Stores the country 2-character code. E.g., US, GB, etc.
3.	**CountryCode3Char (U)**	Characters	CHAR(3)	Yes	**3**	**NOT NULL UNIQUE**	Stores the country 3-character code. E.g., USA, GBR, etc.
4.	**CountryName (U)**	String	VARCHAR(100)	Yes	**100**	**NOT NULL UNIQUE**	Stores the unique full name of the country.

13.Database design Deliverable #5- Physical Model Schema diagram Design :

The Physical Model Schema Diagram for the EZ Auto Rental POS Management System translates the logical database design into an implementation-ready structure within Microsoft SQL Server. It includes detailed specifications such as table names, data types, primary and foreign keys, constraints, and indexing strategies. This deliverable ensures that the database design is not only theoretically sound but also optimized for performance, data integrity, and scalability in a real-world environment.



14.Database Implementation Deliverable #6 –

Development & Implementation:

To implement the database design and bring the logical schema to life, the following MySQL code was written and executed in MySQL Server. This code defines the structure of each table using appropriate data types and constraints such as primary keys, foreign keys, uniqueness, not-null constraints, and check conditions. These ensure data integrity, enforce business rules, and establish proper relationships between entities. The implementation aligns with the original ERD and supports the requirements of the EZ Auto Rental Management System. Below is the MySQL script used to create the necessary tables.

EZRentalDB_Admin_Script to create the database:

```
CREATE DATABASE EZRentalDB;
```

EZrentalDB_Create_Table_Scripts:

#1.

```
 CREATE TABLE COUNTRY
(
   CountryID TINYINT,
   CountryCode2Char CHAR(2) NOT NULL UNIQUE,
   CountryCode3Char CHAR(3) NOT NULL UNIQUE,
   CountryName VARCHAR(100) NOT NULL UNIQUE,

   CONSTRAINT PK_CountryID PRIMARY KEY (CountryID),
   CONSTRAINT CK_CountryIDRange CHECK (CountryID BETWEEN 1 AND 250)
);
```

#2.

```
CREATE TABLE USSTATE
(
   StateID TINYINT,
   StateCode2Char CHAR(2) NOT NULL,
   StateName VARCHAR(50) NOT NULL,

   CONSTRAINT PK_StateID PRIMARY KEY (StateID),
   CONSTRAINT CK_StateIDRange CHECK (StateID BETWEEN 1 AND 75),
   CONSTRAINT UQ_StateCode2Char UNIQUE (StateCode2Char),
   CONSTRAINT UQ_StateName UNIQUE (StateName)
);
```

#3.

```
CREATE TABLE CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY
(
   CreditCardProcessingMerchantServiceCompanyCode TINYINT,
   CreditCardProcessingMerchantServiceCompanyName VARCHAR(50) NOT NULL,

   CONSTRAINT PK_CCProcMerchServCompCode PRIMARY KEY
(CreditCardProcessingMerchantServiceCompanyCode),
   CONSTRAINT CK_CCProcMerchServCompCodeRange CHECK (CreditCardProcessingMerchantServiceCompanyCode
BETWEEN 1 AND 20),
   CONSTRAINT UQ_CCProcMerchServCompName UNIQUE (CreditCardProcessingMerchantServiceCompanyName)
);
```

#4.

CREATE TABLE

CREDITCARDNETWORKCOMPANY (

CreditCardNetworkCompanyCode TINYINT NOT NULL,

CreditCardNetworkCompanyName VARCHAR(50) NOT NULL,

CONSTRAINT PK_CreditCardNetworkCompanyCode PRIMARY KEY (CreditCardNetworkCompanyCode),

CONSTRAINT UQ_CreditCardNetworkCompanyName UNIQUE (CreditCardNetworkCompanyName),

CONSTRAINT CK_CreditCardNetworkCompanyCodeRange CHECK (CreditCardNetworkCompanyCode BETWEEN 1 AND 20)

);

#5.

```
CREATE TABLE

CREDITCARDISSUINGBANK (
   CreditCardIssuingBankCode TINYINT PRIMARY KEY,
   CreditCardIssuingBankName VARCHAR(26) UNIQUE NOT NULL,

   CONSTRAINT CK_CreditCardIssuingBankCodeRange CHECK (CreditCardIssuingBankCode BETWEEN 1 AND 20)
);
```

#6.

```
CREATE TABLE
CREDITCARDCORPORATEMERCHANTBANK (
   CreditCardCorporateMerchantBankCode TINYINT PRIMARY KEY,
   CreditCardCorporateMerchantBankName VARCHAR(30) UNIQUE NOT NULL,

   CONSTRAINT CK_CreditCardCorporateMerchantBankCodeRange CHECK (CreditCardCorporateMerchantBankCode
BETWEEN 1 AND 10)
);
```

#7.

```
CREATE TABLE
CREDITCARD (
   CreditCardNumber VARCHAR(16) PRIMARY KEY,
   CreditCardOwnerName VARCHAR(50) UNIQUE NOT NULL,
   CreditCardProcessingMerchantServiceCompanyCode TINYINT NOT NULL,
   CreditCardNetworkCompanyCode TINYINT NOT NULL,
   CreditCardIssuingBankCode TINYINT NOT NULL,
   CreditCardCorporateMerchantBankCode TINYINT NOT NULL,
   ExpDate DATE NOT NULL,
   AddressLine1 VARCHAR(50) NOT NULL,
   AddressLine2 VARCHAR(50),
   City VARCHAR(50) NOT NULL,
   StateCode CHAR(2) NOT NULL,
   Zipcode VARCHAR(10) NOT NULL,
   Country VARCHAR(100) NOT NULL,
   CreditCardLimit DECIMAL(8,2) NOT NULL,
```

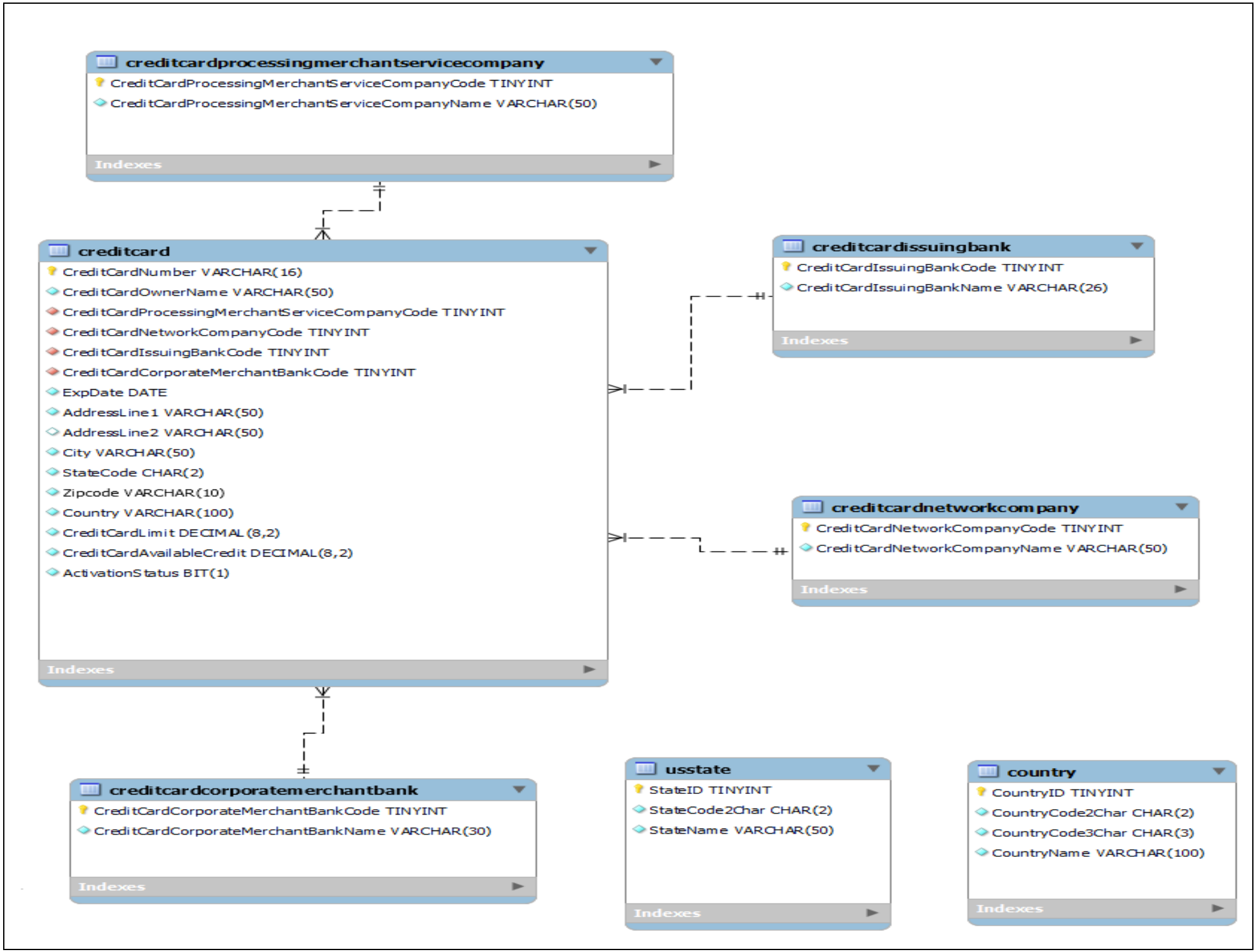
```
    CreditCardAvailableCredit DECIMAL(8,2) NOT NULL,
    ActivationStatus BIT NOT NULL,

    CONSTRAINT fk_CreditCardProcessingMerchantServiceCompany FOREIGN KEY
(CreditCardProcessingMerchantServiceCompanyCode)
        REFERENCES CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY
(CreditCardProcessingMerchantServiceCompanyCode)
        ON DELETE CASCADE ON UPDATE CASCADE,

    CONSTRAINT fk_CreditCardNetworkCompany FOREIGN KEY (CreditCardNetworkCompanyCode)
        REFERENCES CREDITCARDNETWORKCOMPANY (CreditCardNetworkCompanyCode)
        ON DELETE CASCADE ON UPDATE CASCADE,

    CONSTRAINT fk_CreditCardIssuingBank FOREIGN KEY (CreditCardIssuingBankCode)
        REFERENCES CREDITCARDISSUINGBANK (CreditCardIssuingBankCode)
        ON DELETE CASCADE ON UPDATE CASCADE,
```

```
    CONSTRAINT fk_CreditCardCorporateMerchantBank FOREIGN KEY (CreditCardCorporateMerchantBankCode)
        REFERENCES CREDITCARDCORPORATEMERCHANTBANK (CreditCardCorporateMerchantBankCode)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

15.Database Implementation Deliverable #7 –

Implementation Physical Schema Diagram

The EER diagram for the Credit Card Management System illustrates the physical schema of the database implemented in MySQL. It contains 10 tables, including core entities such as CreditCard, CreditCardIssuingBank, CreditCardNetworkCompany, and CreditCardCorporateMerchantBank. These tables are connected through well-defined one-to-many relationships using foreign keys, ensuring data consistency and referential integrity. Each table uses appropriate primary keys, with constraints such as NOT NULL, UNIQUE, and CHECK to enforce business rules. The CreditCard table acts as the central hub, linking to several related entities. The diagram was generated using MySQL Workbench to visually represent the structure and relationships of the database.


creditcardprocessingmerchantservicecompany
CreditCardProcessingMerchantServiceCompanyCode TINYINT
CreditCardProcessingMerchantServiceCompanyName VARCHAR(50)
Indexes
creditcard
CreditCardNumber VARCHAR(16)
CreditCardOwnerName VARCHAR(50)
CreditCardProcessingMerchantServiceCompanyCode TINYINT
CreditCardNetworkCompanyCode TINYINT
CreditCardIssuingBankCode TINYINT
CreditCardCorporateMerchantBankCode TINYINT
ExpDate DATE
AddressLine1 VARCHAR(50)
AddressLine2 VARCHAR(50)
City VARCHAR(50)
StateCode CHAR(2)
Zipcode VARCHAR(10)
Country VARCHAR(100)
CreditCardLimit DECIMAL(8,2)
CreditCardAvailableCredit DECIMAL(8,2)
ActivationStatus BIT(1)
Indexes
creditcardissuingbank
CreditCardIssuingBankCode TINYINT
CreditCardIssuingBankName VARCHAR(26)
Indexes
creditcardnetworkcompany
CreditCardNetworkCompanyCode TINYINT
CreditCardNetworkCompanyName VARCHAR(50)
Indexes
creditcardcorporatemerchantbank
CreditCardCorporateMerchantBankCode TINYINT
CreditCardCorporateMerchantBankName VARCHAR(30)
Indexes
usstate
StateID TINYINT
StateCode2Char CHAR(2)
StateName VARCHAR(50)
Indexes
country
CountryID TINYINT
CountryCode2Char CHAR(2)
CountryCode3Char CHAR(3)
CountryName VARCHAR(100)
Indexes

16.Database Implementation Deliverable #8-

Database validation Unit & Integration Testing :

The Database Validation Unit and Integration Testing phase ensures that the EZ Auto Rental POS Management System performs reliably and accurately according to the defined requirements. During unit testing, each individual table and constraint was tested to confirm correct data entry, enforcement of business rules, and proper handling of edge cases. Integration testing followed, focusing on verifying the relationships and interactions between multiple tables—such as validating foreign key constraints, joins, and cascading updates or deletes. This deliverable confirms that the database functions cohesively as a complete system and is ready for deployment and real-world operations.

DOCUMENT CONTENT INSERT TEMPLATE

Populating The [CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table

The **CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY** table stores information about companies that provide credit card processing services to merchants. Each company is uniquely identified by the CreditCardProcessingMerchantServiceCompanyCode (of type TINYINT), which serves as the table's primary key and is constrained to values between 1 and 20. The CreditCardProcessingMerchantServiceCompanyName column (of type VARCHAR(30)) ensures each company's name is unique and not null, maintaining data integrity and preventing duplicate entries. This table plays a critical role in linking credit card transactions to the processing service providers that facilitate merchant payments.

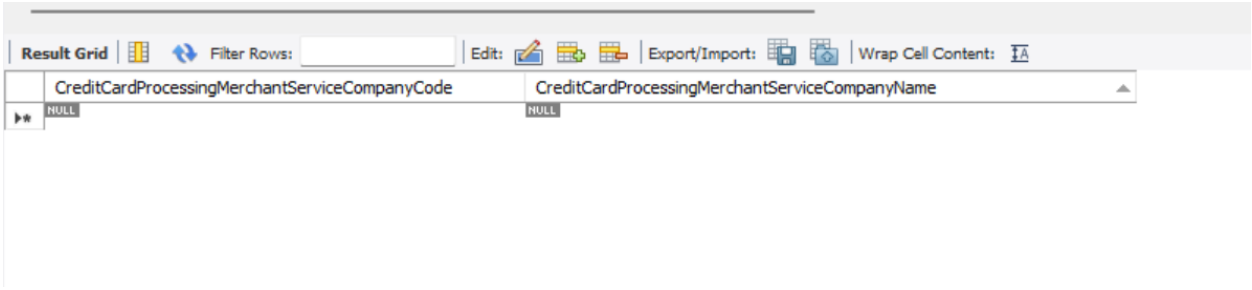
This section shows the content of the **[CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table** before and after populating it with records using **SQL INSERT STATEMENTS**.

Table Before Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table**:

```
SELECT *
FROM [CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY];
```

The current state of the **[CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table** is:



List of SQL INSERT STATEMENTS used to Populate the table:

The following **SQL INSERT Statements** were used to INSERT records to the **[CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table**:

```
INSERT INTO  CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY
(CreditCardProcessingMerchantServiceCompanyCode,
 CreditCardProcessingMerchantServiceCompanyName)
VALUES
(1, 'Stax by Fattmerchant'),
(2, 'Helcim'),
(3, 'Dharma Merchant Service'),
(4,'Payment Depot'),
(5,'National Processing'),
(6,'Block'),
(7,'Intuit Quikbooks'),
(8,'PayPal'),
(9,'Stripe'),
(10,'Flagship Merchant Services'),
(11,'Clover');

select * from creditcardprocessingmerchantservicecompany
order by CreditCardProcessingMerchantServiceCompanyCode ASC;
```

Table After Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table**:

```
SELECT *
```

```
FROM [CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY];
```

The final state of the **[CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table** is:

CreditCardProcessingMerchantServiceCompanyCode	CreditCardProcessingMerchantServiceCompanyName
1	Stax by Fattmerchant
2	Helcim
3	Dharma Merchant Service
4	Payment Depot
5	National Processing
6	Block
7	Intuit Quikbooks
8	PayPal
9	Stripe
10	Flagship Merchant Services
11	Clover
NULL	NULL

DOCUMENT CONTENT INSERT TEMPLATE

Populating The [CREDITCARDNETWORKCOMPANY] Table

The **CREDITCARDNETWORKCOMPANY** table contains data on companies that operate credit card networks, such as Visa or MasterCard. Each company is uniquely identified by the CreditCardNetworkCompanyCode column (of type TINYINT), which serves as the primary key and ensures that values fall within a defined range. The CreditCardNetworkCompanyName column (of type VARCHAR(50)) is both unique and not null, ensuring that each network company's name is distinct and mandatory. This table is essential for mapping each credit card to its respective network provider, enabling accurate processing and reporting within the credit card system.

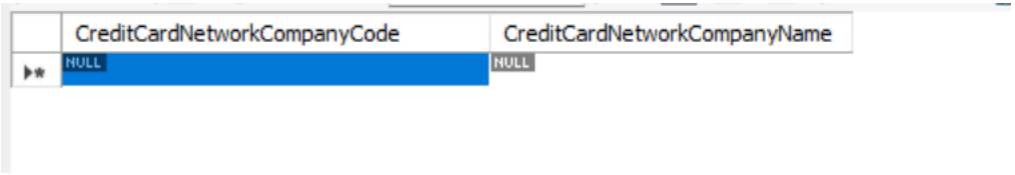
This section shows the content of the **[CREDITCARDNETWORKCOMPANY] Table** before and after populating it with records using **SQL INSERT STATEMENTS**.

Table Before Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARDNETWORKCOMPANY] Table**:

```
SELECT *
FROM [CREDITCARDNETWORKCOMPANY];
```

The current state of the **[CREDITCARDNETWORKCOMPANY] Table** is:



	CreditCardNetworkCompanyCode	CreditCardNetworkCompanyName
▶*	NULL	NULL

List of SQL INSERT STATEMENTS used to Populate the table:

The following **SQL INSERT Statements** were used to INSERT records to the **[CREDITCARDNETWORKCOMPANY] Table**:
#1

```
SELECT * FROM CREDITCARDNETWORKCOMPANY;

#2
INSERT INTO  CREDITCARDNETWORKCOMPANY (CreditCardNetworkCompanyCode, CreditCardNetworkCompanyName)
VALUES
(1, 'America Express'),
(2, 'Visa'),
(3, 'Mastercard'),
(4,'Discover'),
(5,'Diners Club'),
(6,'Interlink'),
(7,'Star'),
(8,'Accel'),
(9,'Interac'),
(10,'Visa ReadyLink'),
(11,'Pulse'),
(12,'JCP (Japan Credit Bureau)'),
(13,'Rupay');


#3
SELECT * FROM CREDITCARDNETWORKCOMPANY
ORDER BY CreditCardNetworkCompanyCode ASC;
```

Table After Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARDNETWORKCOMPANY] Table**:

```
SELECT *
FROM [CREDITCARDNETWORKCOMPANY];
```

The final state of the **[CREDITCARDNETWORKCOMPANY] Table** is:

	CreditCardNetworkCompanyCode	CreditCardNetworkCompanyName
▶	1	America Express
	2	Visa
	3	Mastercard
	4	Discover
	5	Diners Club
	6	Interlink
	7	Star
	8	Accel
	9	Interac
	10	Visa ReadyLink
	11	Pulse
	12	JCP (Japan Credit Bureau)
	13	Rupay
*	NULL	NULL

DOCUMENT CONTENT INSERT TEMPLATE

Populating The [CREDITCARDISSUINGBANK] Table

The **CREDITCARDISSUINGBANK** table holds information about banks that issue credit cards to customers. Each bank is identified by a unique CreditCardIssuingBankCode (of type TINYINT), which serves as the primary key and is constrained to values between 1 and 20 to ensure valid identification. The CreditCardIssuingBankName column (of type VARCHAR(20)) is defined as unique and not null, preventing duplication and ensuring all issuing banks are clearly named. This table plays a key role in associating each credit card with the financial institution responsible for its issuance and account management.

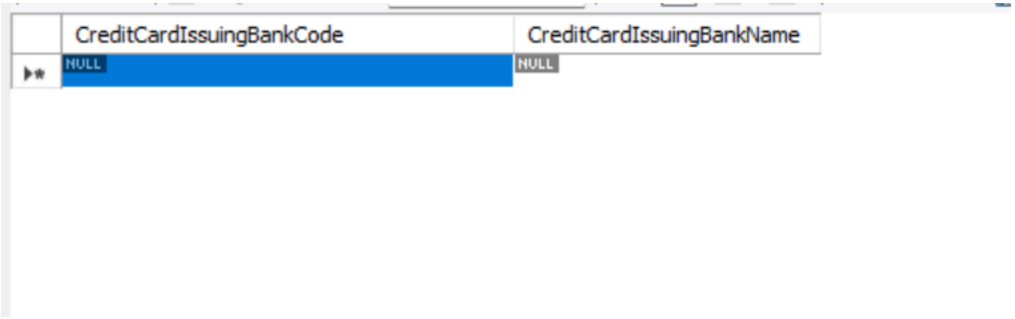
This section shows the content of the **[CREDITCARDISSUINGBANK] Table** before and after populating it with records using **SQL INSERT STATEMENTS**.

Table Before Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARDISSUINGBANK] Table**:

```
SELECT *
FROM [CREDITCARDISSUINGBANK];
```

The current state of the **[CREDITCARDISSUINGBANK] Table** is:



List of SQL INSERT STATEMENTS used to Populate the table:

The following **SQL INSERT Statements** were used to INSERT records to the **[CREDITCARDISSUINGBANK] Table**:

#1
select * from creditcardissuingbank;

#2
INSERT INTO CREDITCARDISSUINGBANK
 (CreditCardIssuingBankCode
, CreditCardIssuingBankName
)
VALUES
(1, 'American Express'),
(2, 'Bank of America'),
(3, 'Barclays'),
(4,'Capital One'),
(5,'Chase'),
(6,'Citi'),
(7,'Discover'),
(8,'Synchrony Bank'),
(9,'U.S. Bank'),
(10,'Wells Fargo');

#3
SELECT * FROM CreditCardIssuingBank
ORDER BY CreditCardIssuingBankCode ASC;

Table After Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARDISSUINGBANK] Table**:

```
SELECT *
FROM [CREDITCARDISSUINGBANK];
```

The final state of the **[CREDITCARDISSUINGBANK] Table** is:

CreditCardIssuingBankCode	CreditCardIssuingBankName
1	American Express
2	Bank of America
3	Barclays
4	Capital One
5	Chase
6	Citi
7	Discover
8	Synchrony Bank
9	U.S. Bank
10	Wells Fargo
NULL	NULL

CreditCardCorporateMerchantBankCode	CreditCardCorporateMerchantBankName
1	Chase
2	Citi
3	Capital One
NULL	NULL

DOCUMENT CONTENT INSERT TEMPLATE

Populating The [CREDITCARD] Table

The **CREDITCARD** table stores detailed information about individual credit cards issued to customers. Each record is uniquely identified by the CreditCardNumber (of type VARCHAR(16)), which serves as the primary key. The table includes key details such as the cardholder's name (CreditCardOwnerName), expiration date (ExpDate), billing address (including AddressLine1, AddressLine2, City, StateCode, Zipcode, and Country), credit limit and available credit (CreditCardLimit, CreditCardAvailableCredit), and activation status (ActivationStatus). It also contains foreign key references to related tables via codes for processing service companies, network companies, issuing banks, and corporate merchant banks, each with a TINYINT type and constrained to valid ranges. This comprehensive table forms the core of the system by linking all aspects of a credit card's data for accurate processing, verification, and reporting.

This section shows the content of the **[CREDITCARD] Table** before and after populating it with records using **SQL INSERT STATEMENTS**.

Table Before Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchant
NULL	NULL	NULL	NULL	NULL	NULL

ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

List of SQL INSERT STATEMENTS used to Populate the table:

The following **SQL INSERT Statements** were used to INSERT records to the **[CREDITCARD] Table**:

#1
```
select * from creditcard;
```

#2
```
INSERT INTO CREDITCARD (
   CreditCardNumber, CreditCardOwnerName, CreditCardProcessingMerchantServiceCompanyCode,
   CreditCardNetworkCompanyCode, CreditCardIssuingBankCode, CreditCardCorporateMerchantBankCode,
   ExpDate, AddressLine1, AddressLine2, City, StateCode, Zipcode, Country,
   CreditCardLimit, CreditCardAvailableCredit, ActivationStatus
)
VALUES
('1111111111111111', 'Joe Smith', 8, 2, 2, 1, '2025-01-01', '111 Jay Street', 'Suit 101', 'Freehold', 'NJ', '17711', 'USA', 3000.00, 1000.00, 1),
('2222222222222221', 'Alex Rogriguez', 7, 6, 4, 2, '2027-02-02', '222 Glenwood Rd', 'Apt 6H', 'Brooklyn', 'NY', '11222', 'USA', 1000.00, 8000.00, 1),
('2222222222222222', 'Alex Rogriguez', 8, 2, 1, 2, '2027-02-02', '222 Glenwood Rd', 'Apt 6H', 'Brooklyn', 'NY', '11222', 'USA', 1000.00, 8000.00, 1),
('2222222222222223', 'Alex Rogriguez', 11, 4, 2, 3, '2027-02-02', '222 Glenwood Rd', 'Apt 6H', 'Brooklyn', 'NY', '11222', 'USA', 1000.00, 8000.00, 1),
('3333333333333333', 'Samuel Peterson', 1, 3, 5, 1, '2024-03-03', '333 5th Avenue', NULL, 'New York', 'NY', '10033', 'USA', 3000.00, 3000.00, 0),
('4444444444444444', 'Mary Johnson', 6, 11, 8, 3, '2029-04-04', '444 Flatland Ave', '3rd Floor', 'Allentown', 'PA', '14344', 'USA', 5000.00, 2000.00, 1),
('5555555555555555', 'Nancy Rivera', 8, 7, 1, 1, '2030-05-05', '555 E45th Street', 'Apt 5A', 'New Haven', 'CT', '12255', 'USA', 3000.00, 500.00, 1);
```

#3
```
SELECT * FROM CREDITCARD
ORDER BY CreditCardNumber ASC;
```

Table After Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The final state of the **[CREDITCARD] Table** is:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCo
1111111111111111	Joe Smith	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	8	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

DOCUMENT CONTENT SELECT TEMPLATE

Searching The [CREDITCARD, CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY,

CREDITCARDNETWORKCOMPANY,

CREDITCARDISSUINGBANK ,

CREDITCARDCORPORATEMERCHANTBANK
]
Table(s) via SQL SELECT Statement

This section shows the test & validates the result of querying the **[TABLE_NAME(S)] Table(s)** using an **SQL SELECT STATEMENT**.

Table Before Execution of SQL SELECT STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[TABLE_NAME(S)] Table(s)**:

```
SELECT *
FROM [TABLE_NAME];
```

The current state of the **[TABLE_NAME] Table** is:

*-- Paste a screenshot of current state or content of the **TABLE_NAME table** after executing the select statement here!*
-- Table should be empty.

*Listing of **SQL SELECT STATEMENT** used to query the table:*

The following **SQL SELECT Statement(s)** was used to ***RETURN*** records from the **[TABLE_NAME(S)] Table(s)**:

```
SELECT
    cc.CreditCardNumber,
    cc.CreditCardOwnerName,
    cc.ExpDate,
    cc.AddressLine1,
    cc.AddressLine2,
    cc.City,
    cc.StateCode,
    cc.ZipCode,
    cc.Country,
    cc.CreditCardLimit,
    cc.CreditCardAvailableCredit,
    cc.ActivationStatus,
    ccpmsc.CreditCardProcessingMerchantServiceCompanyName,
    ccn.CreditCardNetworkCompanyName,
    ccib.CreditCardIssuingBankName,
    cccmb.CreditCardCorporateMerchantBankName
FROM
    CreditCard cc
JOIN
    CreditCardProcessingMerchantServiceCompany ccpmsc
ON
cc.CreditCardProcessingMerchantServiceCompanyCode = ccpmsc.CreditCardProcessingMerchantServiceCompanyCode
JOIN
    CreditCardNetworkCompany ccn
 ON cc.CreditCardNetworkCompanyCode = ccn.CreditCardNetworkCompanyCode
JOIN
    CreditCardIssuingBank ccib
ON
cc.CreditCardIssuingBankCode = ccib.CreditCardIssuingBankCode
JOIN
    CreditCardCorporateMerchantBank cccmb
ON
cc.CreditCardCorporateMerchantBankCode = cccmb.CreditCardCorporateMerchantBankCode;
```

*Results of Execution of **SQL SELECT STATEMENT**:*

Results of the execution of the **SELECT Statement**:

CreditCardNumber	CreditCardOwnerName	ExpDate	AddressLine1	AddressLine2	City	StateCode	ZipCode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus	CreditCardPro
2222222222222223	Alex Rogriguez	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1	Clover
4444444444444444	Mary Johnson	2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1	Block
1111111111111111	Joe Smith	2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1	PayPal
3333333333333333	Samuel Peterson	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0	Stax by Fattme
5555555555555555	Nancy Rivera	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1	PayPal
2222222222222221	Alex Rogriguez	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1	Intuit Quikbook
2222222222222222	Alex Rogriguez	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1	PayPal

CreditCardProcessingMerchantServiceCompanyName	CreditCardNetworkCompanyName	CreditCardIssuingBankName	CreditCardCorporateMerchantBankName
Clover	Discover	Bank of America	Capital One
Block	Pulse	Synchrony Bank	Capital One
PayPal	Visa	Bank of America	Chase
Stax by Fattmerchant	Mastercard	Chase	Chase
PayPal	Star	American Express	Chase
Intuit Quikbooks	Interlink	Capital One	Citi
PayPal	Visa	American Express	Citi

#2

DOCUMENT CONTENT SELECT TEMPLATE

Searching The [CREDITCARD, CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY,

CREDITCARDNETWORKCOMPANY,

CREDITCARDISSUINGBANK ,

CREDITCARDCORPORATEMERCHANTBANK] Table(s) via SQL SELECT Statement

This section shows the test & validates the result of querying the **[TABLE_NAME(S)] Table(s)** using an **SQL SELECT STATEMENT**.

Table Before Execution of SQL SELECT STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[TABLE_NAME(S)] Table(s)**:

```
SELECT *
FROM [TABLE_NAME];
```

The current state of the **[TABLE_NAME] Table** is:

*-- Paste a screenshot of current state or content of the **TABLE_NAME table** after executing the select statement here!*
-- Table should be empty.

*Listing of **SQL SELECT STATEMENT** used to query the table:*

The following **SQL SELECT Statement(s)** was used to ***RETURN*** records from the **[TABLE_NAME(S)] Table(s)**:

```
SELECT
   cc.CreditCardNumber,
   cc.CreditCardOwnerName,
   cc.ExpDate,
   cc.AddressLine1,
   cc.AddressLine2,
   cc.City,
   cc.StateCode,
   cc.ZipCode,
   cc.Country,
   cc.CreditCardLimit,
   cc.CreditCardAvailableCredit,
   cc.ActivationStatus,
   cm.CreditCardProcessingMerchantServiceCompanyName,
   nc.CreditCardNetworkCompanyName,
   ib.CreditCardIssuingBankName,
   mb.CreditCardCorporateMerchantBankName
FROM
   CREDITCARD cc
JOIN
   CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY cm ON cc.CreditCardProcessingMerchantServiceCompanyCode =
cm.CreditCardProcessingMerchantServiceCompanyCode
JOIN
   CREDITCARDNETWORKCOMPANY nc ON cc.CreditCardNetworkCompanyCode = nc.CreditCardNetworkCompanyCode
JOIN
   CREDITCARDISSUINGBANK ib ON cc.CreditCardIssuingBankCode = ib.CreditCardIssuingBankCode
JOIN
   CREDITCARDCORPORATEMERCHANTBANK mb ON cc.CreditCardCorporateMerchantBankCode = mb.CreditCardCorporateMerchantBankCode
WHERE
   cc.CreditCardNumber = '1111111111111111';
```

*Results of Execution of **SQL SELECT STATEMENT**:*

Results of the execution of the **SELECT Statement**:

	CreditCardNumber	CreditCardOwnerName	ExpDate	AddressLine1	AddressLine2	City	StateCode	ZipCode	Country	CreditCardLimit	CreditCardAvailableCredit
▶	1111111111111111	Joe Smith	2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00

ActivationStatus	CreditCardProcessingMerchantServiceCompanyName	CreditCardNetworkCompanyName	CreditCardIssuingBankName	CreditCardCorporateMerchantBankName
1	PayPal	Visa	Bank of America	Chase

#3

DOCUMENT CONTENT SELECT TEMPLATE

Searching The [CREDITCARD] Table(s) via SQL SELECT Statement

This section shows the test & validates the result of querying the **[CREDITCARD] Table(s)** using an **SQL SELECT STATEMENT**.

Table Before Execution of SQL SELECT STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table(s)**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Joe Smith	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	8	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

*Listing of **SQL SELECT STATEMENT** used to query the table:*

The following **SQL SELECT Statement(s)** was used to ***RETURN*** records from the **[CREDITCARD] Table(s)**:

```
select creditcardnumber,creditcardownername,expdate,addressline1,addressline2,city,statecode,zipcode,country,activationstatus
from creditcard
where statecode = 'NY';
```

*Results of Execution of **SQL SELECT STATEMENT**:*

Results of the execution of the **SELECT Statement**:

creditcardnumber	creditcardownername	expdate	addressline1	addressline2	city	statecode	zipcode	country	activationstatus
2222222222222221	Alex Rogriguez	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1
2222222222222222	Alex Rogriguez	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1
2222222222222223	Alex Rogriguez	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1
3333333333333333	Samuel Peterson	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	0

#4

DOCUMENT CONTENT SELECT TEMPLATE

Searching The [CREDITCARD] Table(s) via SQL SELECT Statement

This section shows the test & validates the result of querying the **[CREDITCARD] Table(s)** using an **SQL SELECT STATEMENT**.

Table Before Execution of SQL SELECT STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table(s)**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode
1111111111111111	Joe Smith	8	2
2222222222222221	Alex Rogriguez	7	6
2222222222222222	Alex Rogriguez	8	2
2222222222222223	Alex Rogriguez	11	4
3333333333333333	Samuel Peterson	1	3
4444444444444444	Mary Johnson	6	11
5555555555555555	Nancy Rivera	8	7
NULL	NULL	NULL	NULL

CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
2	1	2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
4	2	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
1	2	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2	3	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
5	1	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
8	3	2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
1	1	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

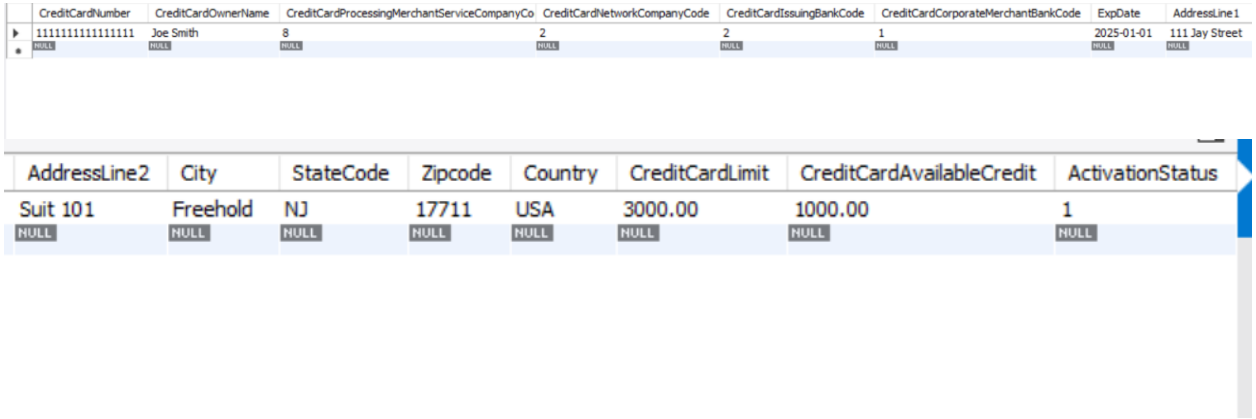
*Listing of **SQL SELECT STATEMENT** used to query the table:*

The following **SQL SELECT Statement(s)** was used to ***RETURN*** records from the **[CREDITCARD] Table(s)**:

```
select * from creditcard where CreditCardNumber = '1111111111111111';
```

*Results of Execution of **SQL SELECT STATEMENT**:*

Results of the execution of the **SELECT Statement**:



CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1
1111111111111111	Joe Smith	8	2	2	1	2025-01-01	111 Jay Street
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#1DOCUMENT CONTENT UPDATE TEMPLATE

Updating Record(s) in the [CREDITCARD] Table via SQL UPDATE Statement

This section shows the results of testing and validation of **[CREDITCARD] Table** *before* and *after* **QUERYING** the **TABLE** using an **SQL UPDATE STATEMENT**.

Table Before Execution of SQL UPDATE STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Joe Smith	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	8	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Listing of the SQL UPDATE STATEMENT used to update the table:

The following **SQL UPDATE Statements** was used to ***UPDATE*** records in the **[CREDITCARD] Table**:

```
UPDATE creditcard
```

```
SET CreditcardOwnerName = 'Ahmad'
WHERE CreditCardNumber = '1111111111111111';
```

Table After Execution of SQL UPDATE STATEMENT:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The final state of the **CREDITCARD Table** is:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	8	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#2

DOCUMENT CONTENT UPDATE TEMPLATE

Updating Record(s) in the [CREDITCARD] Table via SQL UPDATE Statement

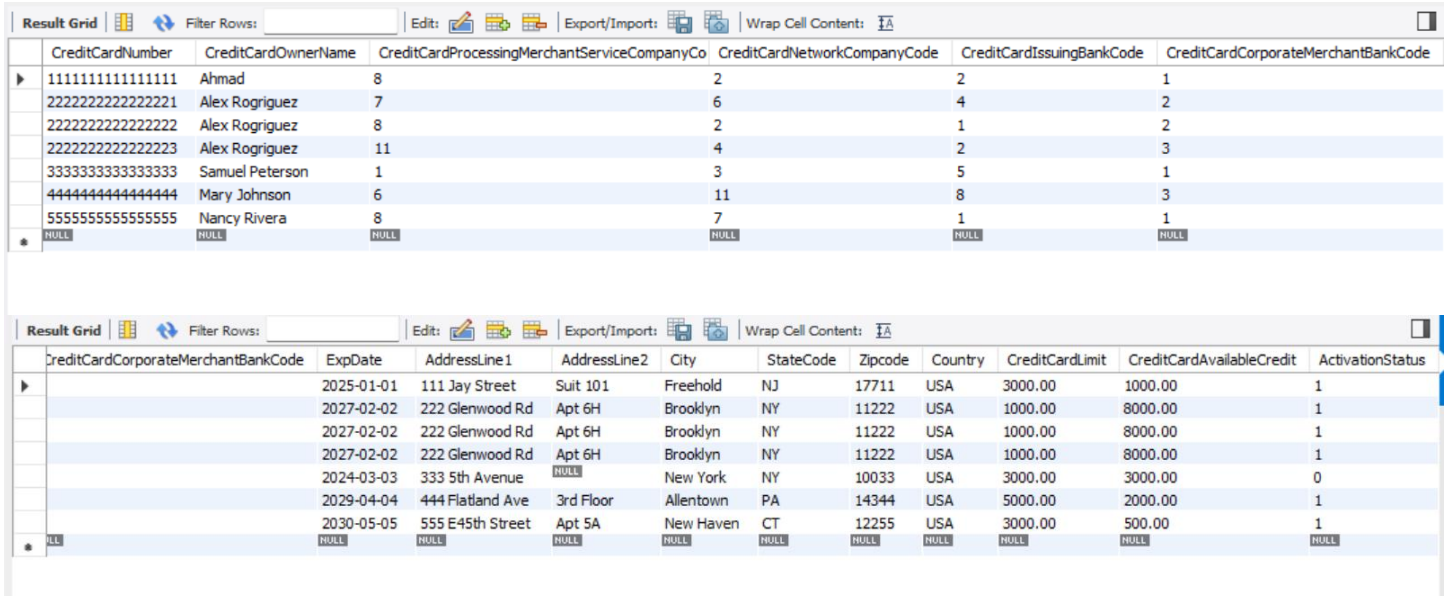
This section shows the results of testing and validation of **[CREDITCARD] Table** *before* and *after* **QUERYING** the **TABLE** using an **SQL UPDATE STATEMENT**.

Table Before Execution of SQL UPDATE STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	8	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	111 Jay Street	Suit 101	Freehold	NJ	17711	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

*Listing of the **SQL UPDATE STATEMENT** used to update the table:*

The following **SQL UPDATE Statements** was used to ***UPDATE*** records in the **[CREDITCARD] Table**:

```
UPDATE creditcard
SET Addressline1 ='Shopner Ghor',
        addressline2 ='California',
   City ='California',
   StateCode = 'CA',
   ZipCode = '11234',
   Country = 'USA'
WHERE CreditCardNumber = '1111111111111111';
```

*Table After Execution of **SQL UPDATE STATEMENT**:*

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The final state of the **CREDITCARD Table** is:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	8	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	Shopner Ghor	California	California	CA	11234	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#3

DOCUMENT CONTENT UPDATE TEMPLATE

Updating Record(s) in the [CREDITCARD & CREDITCARDPROCESSINGSERVICECOMPANY] Table via SQL UPDATE Statement

This section shows the results of testing and validation of **[CREDITCARD] Table** *before* and *after* **QUERYING** the **TABLE** using an **SQL UPDATE STATEMENT**.

Table Before Execution of SQL UPDATE STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[TABLE_NAME] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:

	CreditCardNumber	CreditCardProcessingMerchantServiceCompanyCo
▶	3333333333333333	1
	4444444444444444	6
	2222222222222221	7
	1111111111111111	8
	2222222222222222	8
	5555555555555555	8
	2222222222222223	11
*	NULL	NULL

Listing of the SQL UPDATE STATEMENT used to update the table:

The following **SQL UPDATE Statements** was used to ***UPDATE*** records in the **[TABLE_NAME] Table**:
#

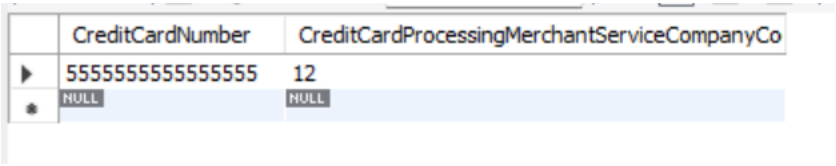
```
INSERT INTO CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY (
   CreditCardProcessingMerchantServiceCompanyCode,
   CreditCardProcessingMerchantServiceCompanyName
)
VALUES (12, 'Square');
SELECT CreditCardNumber, CreditCardProcessingMerchantServiceCompanyCode
FROM CREDITCARD;

#
UPDATE CREDITCARD
SET CreditCardProcessingMerchantServiceCompanyCode = 12
WHERE CreditCardNumber = '5555555555555555';
#
SELECT CreditCardNumber, CreditCardProcessingMerchantServiceCompanyCode
FROM CREDITCARD
WHERE CreditCardNumber = '5555555555555555';
#
select * from CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY;
#
select * from creditcard;
```

Table After Execution of SQL UPDATE STATEMENT:

The following **SQL SELECT Statement** was used to list the content of the **[TABLE_NAME] Table**:

```
SELECT *
FROM [CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY];
```

The final state of the **CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY Table** is:



CreditCardNumber	CreditCardProcessingMerchantServiceCompanyCo
5555555555555555	12
NULL	NULL

#1

DOCUMENT CONTENT DELETE TEMPLATE

Deleting Record(s) from the CREDITCARD Table via SQL DELETE Statement

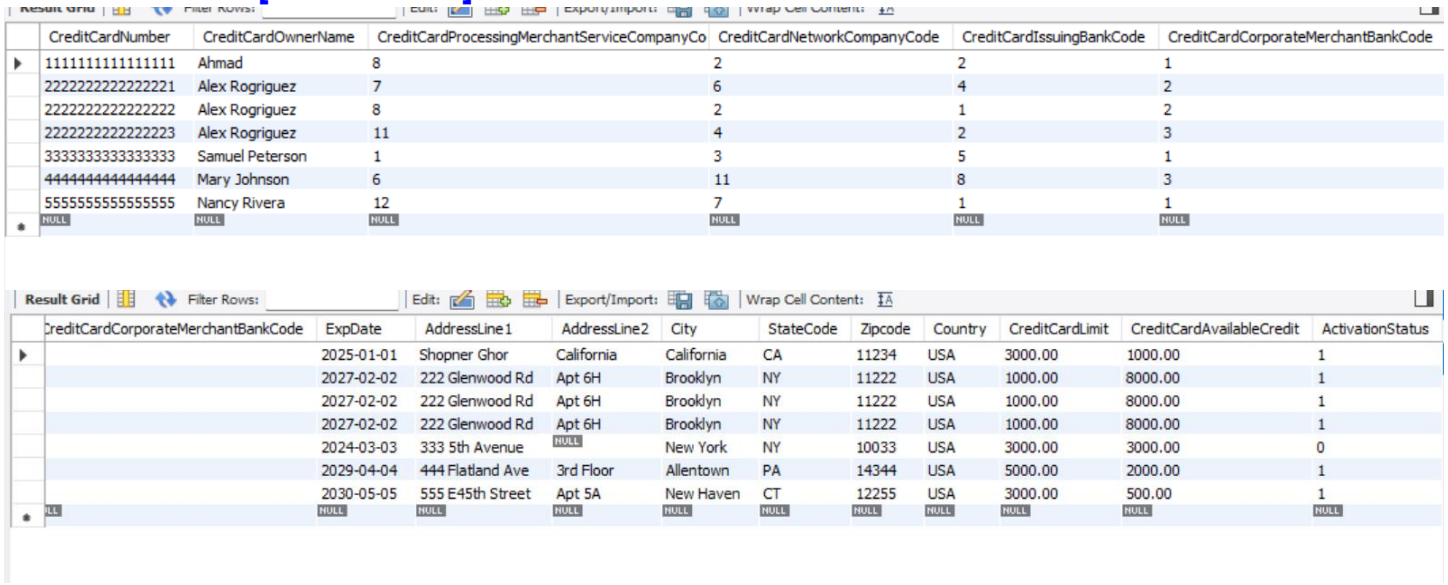
This section shows the results of testing and validation of **[CREDITCARD] Table** ***before*** and ***after*** **QUERYING** the **TABLE** using an **SQL DELETE STATEMENT**.

Table Before Execution of SQL DELETE STATEMENT for Testing & Validation:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The current state of the **[CREDITCARD] Table** is:



CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
4444444444444444	Mary Johnson	6	11	8	3
5555555555555555	Nancy Rivera	12	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	Shopner Ghor	California	California	CA	11234	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2029-04-04	444 Flatland Ave	3rd Floor	Allentown	PA	14344	USA	5000.00	2000.00	1
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

*Listing of **SQL DELETE STATEMENT** used to delete record(s) from the table:*

The following **SQL DELETE Statements** was used to ***DELETE*** record(s) in the **[CREDITCARD] Table**:
#

```
DELETE
FROM creditcard
WHERE CREDITCARDNUMBER = '4444444444444444';
```

*Table After Execution of **SQL DELETE STATEMENT**:*

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD] Table**:

```
SELECT *
FROM [CREDITCARD];
```

The final state of the **CREDITCARD Table** is:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
5555555555555555	Nancy Rivera	12	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	Shopner Ghor	California	California	CA	11234	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#2

DOCUMENT CONTENT DELETE TEMPLATE

Deleting Record(s) from the CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY Table via SQL DELETE Statement

To delete a record from the CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY table without automatically deleting its related child records in the CREDITCARD table—despite the ON DELETE CASCADE constraint implemented in Sprint #3—a workaround was necessary. Since altering the table structure was not permitted, we temporarily disabled foreign key checks using the MySQL-specific command SET FOREIGN_KEY_CHECKS = 0. This allowed us to delete the parent record while preserving the child records. After executing the delete statement based on the primary key of the target record, we re-enabled foreign key checks with SET FOREIGN_KEY_CHECKS = 1. This approach successfully circumvented the cascading delete behavior without compromising the existing database schema.

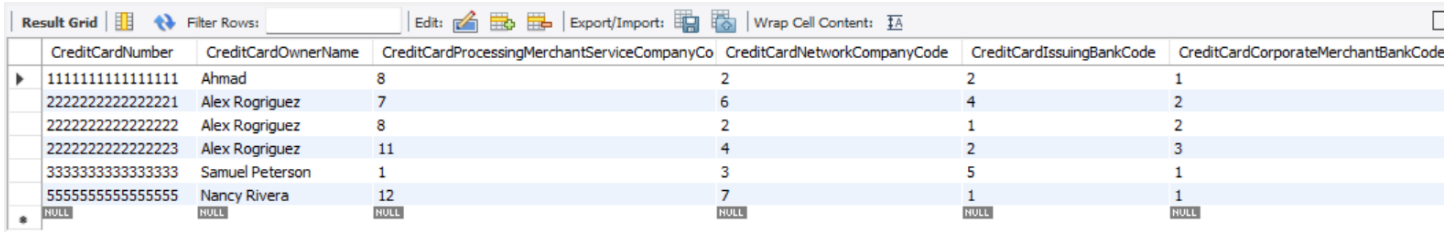
This section shows the results of testing and validation of **[CREDITCARD] Table** ***before*** and ***after*** **QUERYING** the **TABLE** using an **SQL DELETE STATEMENT**.

Table Before Execution of SQL DELETE STATEMENT for Testing & Validation:

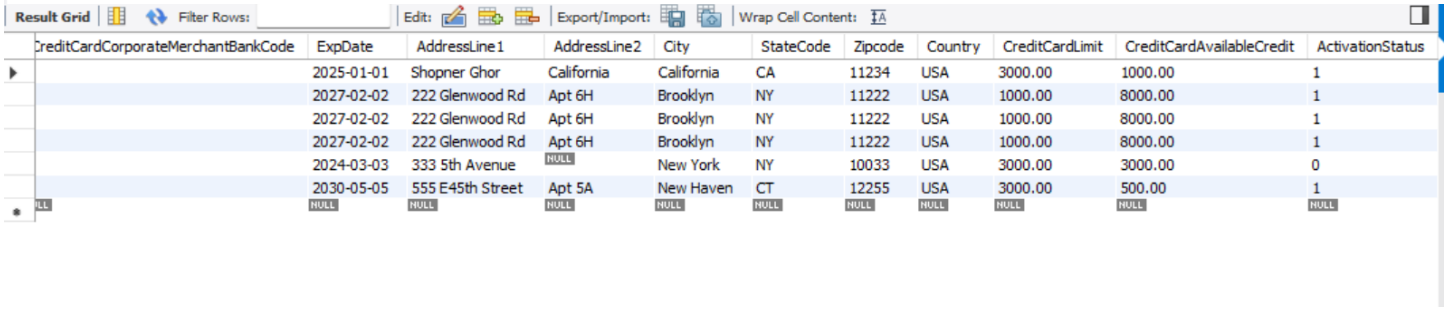
The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table**:

```
SELECT *
FROM [CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY];
```

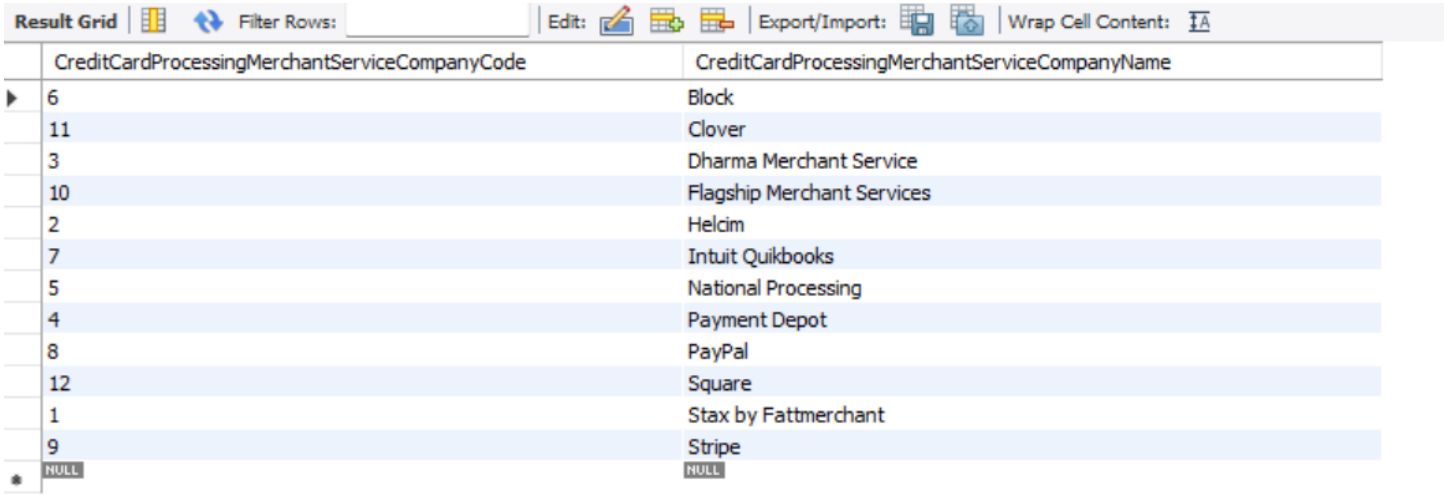
The current state of the **[CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table** is:



CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
5555555555555555	Nancy Rivera	12	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL



reditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	Shopner Ghor	California	California	CA	11234	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
LL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



CreditCardProcessingMerchantServiceCompanyCode	CreditCardProcessingMerchantServiceCompanyName
6	Block
11	Clover
3	Dharma Merchant Service
10	Flagship Merchant Services
2	Helcim
7	Intuit Quikbooks
5	National Processing
4	Payment Depot
8	PayPal
12	Square
1	Stax by Fattmerchant
9	Stripe
NULL	NULL

*Listing of **SQL DELETE STATEMENT** used to delete record(s) from the table:*

The following **SQL DELETE Statements** was used to ***DELETE*** record(s) in the **[CREDITCARD] Table**:
#1

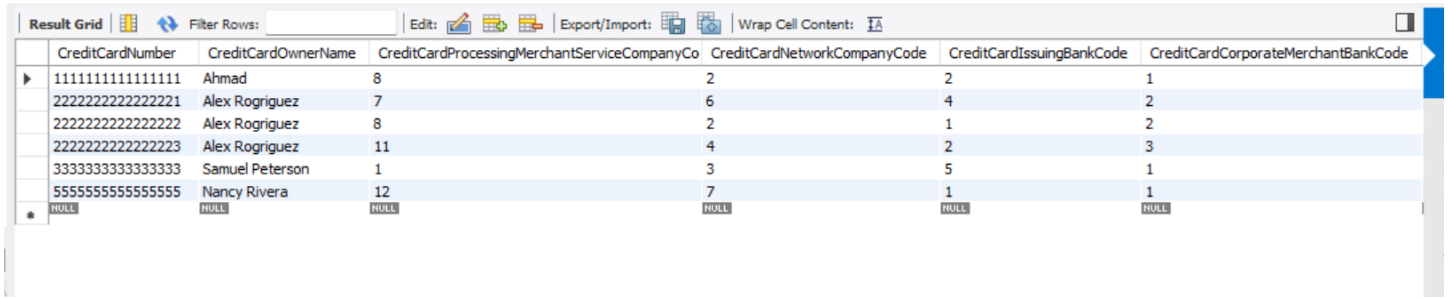
```
SET FOREIGN_KEY_CHECKS = 0;
#2

DELETE FROM CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY
WHERE CreditCardProcessingMerchantServiceCompanyCode = 12;

SET FOREIGN_KEY_CHECKS = 1;
#3
select * from creditcard;
#4
Select * from CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY;
```

Table After Execution of SQL DELETE STATEMENT:

The following **SQL SELECT Statement** was used to list the content of the **[CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY] Table**:

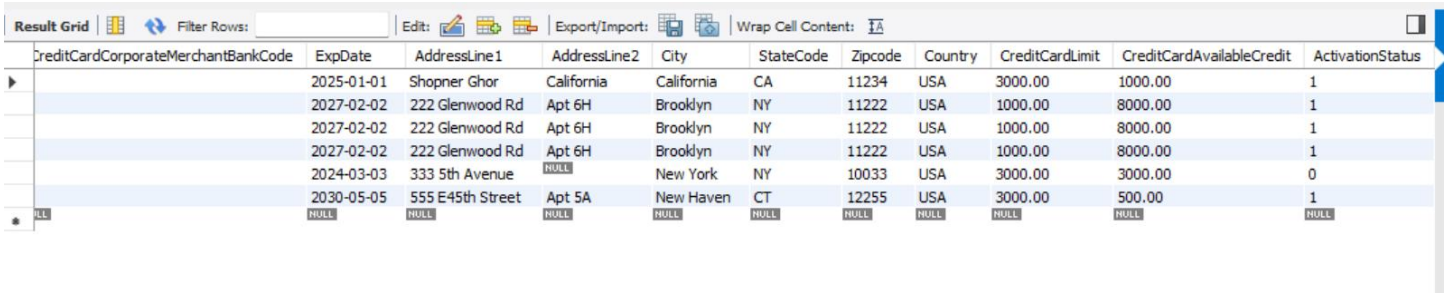
```
SELECT *
FROM [CREDITCARD];
```

The final state of the **CREDITCARD & CREDITCARDPROCESSINGMERCHANTSERVICECOMPANY Table** is:



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardNumber	CreditCardOwnerName	CreditCardProcessingMerchantServiceCompanyCo	CreditCardNetworkCompanyCode	CreditCardIssuingBankCode	CreditCardCorporateMerchantBankCode
1111111111111111	Ahmad	8	2	2	1
2222222222222221	Alex Rogriguez	7	6	4	2
2222222222222222	Alex Rogriguez	8	2	1	2
2222222222222223	Alex Rogriguez	11	4	2	3
3333333333333333	Samuel Peterson	1	3	5	1
5555555555555555	Nancy Rivera	12	7	1	1
NULL	NULL	NULL	NULL	NULL	NULL



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

CreditCardCorporateMerchantBankCode	ExpDate	AddressLine1	AddressLine2	City	StateCode	Zipcode	Country	CreditCardLimit	CreditCardAvailableCredit	ActivationStatus
	2025-01-01	Shopner Ghor	California	California	CA	11234	USA	3000.00	1000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2027-02-02	222 Glenwood Rd	Apt 6H	Brooklyn	NY	11222	USA	1000.00	8000.00	1
	2024-03-03	333 5th Avenue	NULL	New York	NY	10033	USA	3000.00	3000.00	0
	2030-05-05	555 E45th Street	Apt 5A	New Haven	CT	12255	USA	3000.00	500.00	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

CreditCardProcessingMerchantServiceCompanyCode	CreditCardProcessingMerchantServiceCompanyName
6	Block
11	Clover
3	Dharma Merchant Service
10	Flagship Merchant Services
2	Helcim
7	Intuit Quikbooks
5	National Processing
4	Payment Depot
8	PayPal
1	Stax by Fattmerchant
9	Stripe
NULL	NULL

DOCUMENT CONTENT IMPORT TEMPLATE (USSTATE & COUNTRY TABLES ONLY)

Populating The [US STATE] Table

This section shows the content of the **[US STATE] Table** before and after ***IMPORTING*** with records from a **Comma-Separated-Values (CSV) FILE**.
This section presents the contents of the **US STATE** table before and after importing records from a Comma-Separated Values (CSV) file. Initially, the table contained a limited number of records, but after the import, it was successfully updated with all 56 U.S. states. The comparison highlights how the import process enriched the dataset with complete and accurate information.

Table Before Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[US STATE] Table**:

```
SELECT *
FROM USSTATE;
```

The current state of the **[US STATE] Table** is:

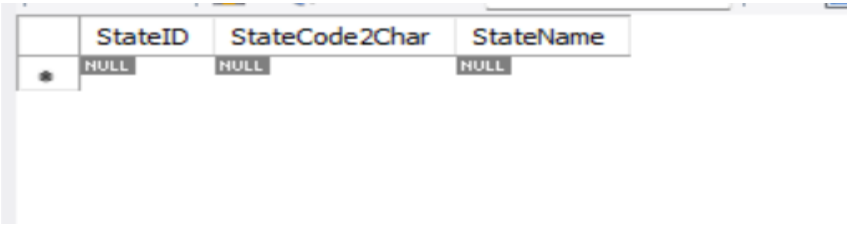


Table After Importing from a from a Comma-Separated-Values (CSV) FILE:

The following **SQL SELECT Statement** was used to list the content of the **[US STATE] Table**:
Csv file imported properly:

Import Results

File D:\spring 2024\.net\project\Sprint4\CST4708_SPRINT4\CST4708_SPRINT4\USState_Country_CSV_Import_Files\USState_Country_CSV_Import_Files\USState.csv was imported in 0.263 s

Table ezrentaldb.usstate has been used

56 records imported

```
SELECT *
FROM USSTATE;
```

The following is a ***SNAPSHOT*** of the final state of the **[US STATE] Table** since the content of the table. Since the table contains a lot of records and is very large, we are only providing a SNAPSHOT as proof:

Result Grid | Filter Rows: | Edit:

StateID	StateCode2Char	StateName
1	AL	ALABAMA
2	AK	ALASKA
3	AS	AMERICAN SAMOA
4	AZ	ARIZONA
5	AR	ARKANSAS
6	CA	CALIFORNIA
7	CO	COLORADO
8	CT	CONNECTICUT
9	DE	DELAWARE
10	DC	DISTRICT OF COLUMBIA
11	FL	FLORIDA
12	GA	GEORGIA
13	GU	GUAM
14	HI	HAWAII
15	ID	IDAHO
16	IL	ILLINOIS
17	IN	INDIANA
18	IA	IOWA
19	KS	KANSAS
20	KY	KENTUCKY
21	LA	LOUSIANA
22	ME	MAINE
23	MP	MARIANA ISLANDS
24	MD	MARYLAND
25	MA	MASSACHUSETTS
26	MI	MICHIGAN
27	MN	MINNESOTA
28	MS	MISSISSIPPI
29	MO	MISSOURI
30	MT	MONTANA
31	NE	NEBRASKA
32	NV	NEVADA
33	NH	NEW HAMPSHIRE
34	NJ	NEW JERSEY
35	NM	NEW MEXICO
36	NY	NEW YORK
37	NC	NORTH CAROLINA
38	ND	NORTH DAKOTA
39	OH	OHIO
40	OK	OKLAHOMA
41	OR	OREGON
42	PA	PENNSYLVANIA
43	PR	PUERTO RICO
44	RI	RHODE ISLAND
45	SC	SOUTH CAROLINA
46	SD	SOUTH DAKOTA

DOCUMENT CONTENT IMPORT TEMPLATE (USSTATE & COUNTRY TABLES ONLY)

Populating The [COUNTRY] Table

The following SQL SELECT statement was used to display the contents of the **COUNTRY** table after importing data from a CSV file. This output confirms that the table was successfully populated with all 240 country records, ensuring completeness and accuracy in the dataset.

This section shows the content of the **[COUNTRY] Table** before and after ***IMPORTING*** with records from a **Comma-Separated-Values (CSV) FILE**.

Table Before Execution of SQL INSERT STATEMENTS:

The following **SQL SELECT Statement** was used to list the content of the **[COUNTRY] Table**:

```
SELECT *
FROM COUNTRY;
```

The current state of the **[COUNTRY] Table** is:

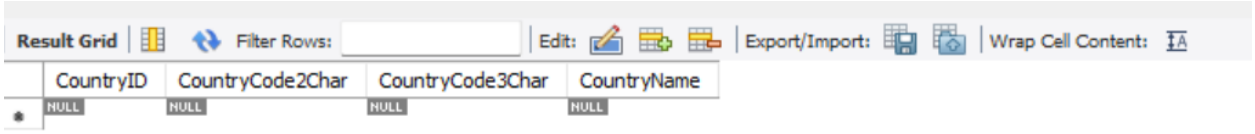
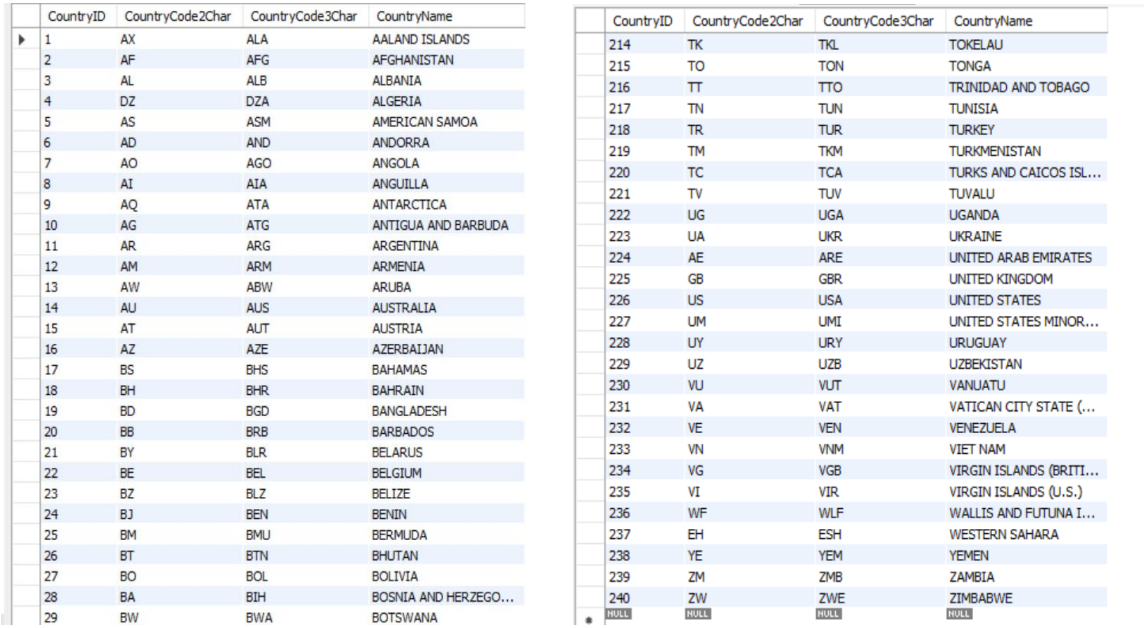


Table After Importing from a from a Comma-Separated-Values (CSV) FILE:

The following **SQL SELECT Statement** was used to list the content of the **[COUNTRY] Table**:

```
SELECT *
FROM COUNTRY;
```

The following is a ***SNAPSHOT*** of the final state of the **[COUNTRY] Table** since the content of the table. Since the table contains a lot of records and is very large, we are only providing a SNAPSHOT as proof:



CountryID	CountryCode2Char	CountryCode3Char	CountryName
1	AX	ALA	AALAND ISLANDS
2	AF	AFG	AFGHANISTAN
3	AL	ALB	ALBANIA
4	DZ	DZA	ALGERIA
5	AS	ASM	AMERICAN SAMOA
6	AD	AND	ANDORRA
7	AO	AGO	ANGOLA
8	AI	AIA	ANGUILLA
9	AQ	ATA	ANTARCTICA
10	AG	ATG	ANTIGUA AND BARBUDA
11	AR	ARG	ARGENTINA
12	AM	ARM	ARMENIA
13	AW	ABW	ARUBA
14	AU	AUS	AUSTRALIA
15	AT	AUT	AUSTRIA
16	AZ	AZE	AZERBAIJAN
17	BS	BHS	BAHAMAS
18	BH	BHR	BAHRAIN
19	BD	BGD	BANGLADESH
20	BB	BRB	BARBADOS
21	BY	BLR	BELARUS
22	BE	BEL	BELGIUM
23	BZ	BLZ	BELIZE
24	BJ	BEN	BENIN
25	BM	BMU	BERMUDA
26	BT	BTN	BHUTAN
27	BO	BOL	BOLIVIA
28	BA	BIH	BOSNIA AND HERZEGO...
29	BW	BWA	BOTSWANA

CountryID	CountryCode2Char	CountryCode3Char	CountryName
214	TK	TKL	TOKELAU
215	TO	TON	TONGA
216	TT	TTO	TRINIDAD AND TOBAGO
217	TN	TUN	TUNISIA
218	TR	TUR	TURKEY
219	TM	TKM	TURKMENISTAN
220	TC	TCA	TURKS AND CAICOS ISL...
221	TV	TUV	TUVALU
222	UG	UGA	UGANDA
223	UA	UKR	UKRAINE
224	AE	ARE	UNITED ARAB EMIRATES
225	GB	GBR	UNITED KINGDOM
226	US	USA	UNITED STATES
227	UM	UMI	UNITED STATES MINOR...
228	UY	URY	URUGUAY
229	UZ	UZB	UZBEKISTAN
230	VU	VUT	VANUATU
231	VA	VAT	VATICAN CITY STATE (...
232	VE	VEN	VENEZUELA
233	VN	VNM	VIET NAM
234	VG	VGB	VIRGIN ISLANDS (BRITI...
235	VI	VIR	VIRGIN ISLANDS (U.S.)
236	WF	WLF	WALLIS AND FUTUNA I...
237	EH	ESH	WESTERN SAHARA
238	YE	YEM	YEMEN
239	ZM	ZMB	ZAMBIA
240	ZW	ZWE	ZIMBABWE
NULL	NULL	NULL	NULL

17.Conclusion:

The EZ Auto Rental Management System represents a significant advancement in the digital transformation of rental operations. Designed with scalability, efficiency, and user experience at its core, the system delivers a unified platform that empowers car rental companies, fleet managers, and automotive dealerships to enhance productivity, reduce operational friction, and elevate customer satisfaction.

By integrating key functionalities—such as reservation management, real-time vehicle tracking, centralized customer data, automated billing, and analytics—the system optimizes core processes and supports data-driven strategies. Its modular design and use of modern technologies ensure adaptability to evolving business needs and technological landscapes.

The hybrid agile-waterfall development approach enabled consistent feedback loops, fostering a system that aligns closely with end-user requirements and stakeholder expectations. Additionally, the chosen technology stack provides flexibility, reliability, and scalability for both current operations and future expansion.

Looking forward, the EZ Auto Rental Management System is well-positioned for continuous growth and innovation. Planned enhancements, including mobile application support, AI-powered forecasting, and global market reach, will further increase its value proposition. These advancements will not only support proactive business decision-making but also solidify the system's role as a strategic enabler in the competitive rental and mobility services sector.

In essence, this system is more than a database project—it is a forward-thinking solution that lays the foundation for smarter, more profitable rental operations in a rapidly evolving industry.