A logo of university of manila

AI-generated content may be incorrect.A logo for a computer science

AI-generated content may be incorrect.

**The University of Manila**

College of Computer Science

*546 Dr. Mariano V. De los Santos St., Manila City, Metro Manila,*

*Philippines*

735-5085 735-5084 735-5098 735- 5094

FAX: 735-5089 521-8773

Email: Admin@um.edu.ph

Website: [www.um.edu.ph](http://www.um.edu.ph)

**Implementation of a Parking Management System for ParkZone Solutions**

In Partial Fulfillment of the Requirements

In Human and Computer Interaction 101

For the Degree of Bachelor of Science in Computer Science

A Thesis Presented to:

**Prof. Rogelio T. Plaza Jr.**

Faculty of Computer Science

Presented by:

Jaculba, Joshua N.

Navarro, John Paul D.

**Table of Contents**

Acknowledgements ……………………………………………………………………………….1

Introduction ...……………………………………………………………………………………..2

History …………………………………………………………………………………………….3

Organizational Chart ……………………………………………………………………………...4

Background of the Study …………………………………………………………………………5

Statement of the Problem …………………………………………………………………………6

Objectives of the Study …………………………………………………………………………...7

Review of Related Literature ……………………………………………………………………..8

Old System Screenshot…………………………………………………………………………..10

Old System Flowchart …………………………………………………………………………...11

Proposed System Flowchart ……………………………………………………………………..12

Data Flow Diagram ……………………………………………………………………………...22

Entity Relationships Diagram ….....……………………………………………………………..23

Definition of Terms ……………………………………………………………………………...24

System Requirements ……………………………………………………………………………25

Software and Hardware Used …………………………………………………………………...26

Cost and Benefits…………... …………………………………………………………………...27

Summary ………………………………………………………………………………………...28

Recommendation ………………………………………………………………………………..29

Data Gathering …………………………………………………………………………………..30

Conclusion ………………………………………………………………………………………31

New System Screenshot …………………………………………………………………………32

Sample Receipt ………………………………………………………………………………….36

Installation Guide………………………………………………………………………………...38

Source Code……………………………………………………………………………………...41

Team Composition……………………………………………………………………………...154

**Acknowledgement**

We would like to thank everyone who helped and supported us in completing our thesis. This project became possible because of the guidance and encouragement we received from many people.

We are very thankful to Kristine Joy M. Villanueva for allowing us to interview them and share their knowledge and experiences. Their time and cooperation gave us important information that greatly helped in the development of our study.

First, we give our sincere thanks to our adviser, Professor Rogelio T. Plaza Jr. for patiently guiding us and giving helpful advice throughout this study. His/her support and suggestions were very important in finishing this project.

We would also like to express our appreciation to various YouTube content creators whose educational videos provided us with additional knowledge, tutorials, and insights that guided us in understanding certain concepts and techniques used in our system.

To our classmates, peers, and friends, thank you for the encouragement, teamwork, and support you gave us during this journey.

Most of all, we are deeply grateful to our families for their love, patience, and understanding. Their support gave us the strength and motivation to finish this thesis.

Lastly, we thank Almighty God for giving us wisdom, guidance, and the opportunity to accomplish this study.

**Introduction**

For many years, ParkZone Solutions, located at 781 M.V. Delos Santos St, Sampaloc, Manila, 1008 Metro Manila, has relied on traditional and manual methods to manage its parking operations. From recording vehicle entries and exits, monitoring available slots, handling payments, and managing customer records, the processes have largely depended on pen-and-paper logging or basic spreadsheets. While these methods have worked in the past, they are increasingly inefficient and prone to errors in today’s fast-paced environment where customers demand convenience, speed, and accuracy.

Manual parking management often results in issues such as lost or inaccurate records, delayed retrieval of parking information, unorganized slot allocation, and inefficient payment processing. Customers also experience long queues and disputes regarding parking duration since there is no unique system-generated identifier to verify time-in and time-out. Another challenge is the lack of accountability among staff, as anyone can access or update logbooks without leaving a trace.

In addition, ParkZone Solutions current pricing system is inconsistent. The standard rate is ₱50 per hour, while the late fee is ₱120 per hour. However, this rate is not strictly implemented, as charges often fluctuate depending on the situation. Furthermore, the existing manual process makes it difficult to track late durations accurately. To improve fairness and precision, the new system will introduce an automated computation feature that charges customers even for partial overtime at a rate of ₱2 per minute ensuring transparent and accurate fee calculations.

To address these problems, this project proposes the development of a Parking Management System for ParkZone Solutions that will automate vehicle entry and exit tracking, parking slot allocation, and fee’s computation while introducing a QR-based logging system to ensure accurate time records. The system will also implement secure staff authentication to strengthen accountability and prevent unauthorized access. By providing automated, transparent, and reliable operations, ParkZone Solutions can reduce human errors, ensure fair and consistent pricing, improve customer satisfaction, and establish a more efficient foundation for long-term growth in Metro Manila.

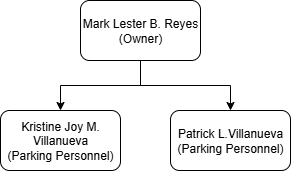
**History**

ParkZone Solutions, located at 781 M.V. Delos Santos St, Sampaloc, Manila, 1008 Metro Manila, has been a familiar establishment in the area for several years, serving as a convenient parking space for residents, commuters, and visitors in the busy district of Sampaloc. Established in the mid-2023, ParkZone Solutions was originally developed to provide a secure and accessible parking facility in a location where traffic congestion and limited parking options were becoming major concerns. Its strategic location near commercial establishments, schools, and residential areas quickly made it a reliable choice for vehicle owners in need of safe and affordable parking.

Over time, ParkZone Solutions has built a reputation for offering dependable parking services, ensuring that vehicles are accommodated and guarded within its premises. However, much of its operations have continued to rely on traditional methods, manual recording of vehicle entries and exits, handwritten tickets, and basic logbooks for tracking availability and payments. While this approach has allowed ParkZone Solutions to function effectively in its early years, it has also introduced challenges such as record inaccuracies, inefficient monitoring of slots, and delays during peak hours.

As the demand for parking spaces continues to grow alongside the rapid urbanization of Manila, ParkZone Solutions recognizes the need to modernize its operations. The growing expectations of vehicle owners for faster, more convenient, and secure services highlight the importance of adapting to digital solutions. Today, ParkZone Solutions stands at a turning point as it seeks to transition from manual processes to an automated Parking Management System. This modernization aims to preserve the facility’s commitment to accessibility and security while addressing the challenges of inefficiency and customer dissatisfaction. Through this shift, ParkZone Solutions envisions becoming not only a reliable parking provider but also a modern, technology-driven facility capable of meeting the evolving needs of the community.

**Organizational Chart**

****

**Background of the Study**

In today’s fast-paced and technology-driven world, the need for efficient systems has become more important in different industries, especially in transportation and parking management. With the continuous growth of urban populations and the increasing number of vehicles on the road, finding safe, organized, and reliable parking has become a common challenge for many drivers. Customers now expect parking facilities not only to provide secure spaces but also to deliver fast, convenient, and hassle-free services.

At present, ParkZone Solutions, located at 781 M.V. Delos Santos St, Sampaloc, Manila, still uses a manual system for handling daily parking operations. Vehicle entries and exits are written by hand, slot availability is checked by observation, and payments are recorded using logbooks. While this method has worked for some time, it often leads to errors, delays, and difficulties in generating accurate records. These issues result in long waiting times, poor slot management, and customer dissatisfaction problems that could be solved with a more modern and automated approach.

In line with its Mission to provide customers with safe and convenient spaces and its Vision of becoming a trusted and reliable parking facility for the community, ParkZone Solutions recognizes the importance of innovation and improvement in its operations. Guided by the Motto, *“Your Space, Your Safety,”* the facility aims to provide efficient and secure services.

Because of these challenges, the researchers decided to study and develop a Parking Management System for ParkZone Solutions. The main purpose of this project is to automate important tasks such as vehicle tracking, slot monitoring, fee computation, payment recording, and report generation. By introducing this system, ParkZone Solutions can reduce human error, speed up transactions, improve customer service, and ensure more organized and reliable operations.

This study is important because it will not only improve ParkZone Solutions daily operations but also show how digital solutions can transform traditional businesses. With this system, ParkZone Solutions can keep up with modern demands, provide better service to customers, and establish itself as a more efficient and technology-driven parking facility in Metro Manila.

**Statement of the Problem**

ParkZone Solutions current system faces several challenges that affect efficiency and accuracy. These issues create delays, inconsistencies, and inconvenience for both staff and customers, highlighting the need for improvement:

* Inaccurate Records
  + Vehicle entries, exits, and payments are written manually, which can lead to errors, missing data, or inconsistencies.
* Unorganized Slot Allocation
  + Parking slots are assigned based only on staff observation, which often causes confusion, double allocation, and inefficient use of space.
* Inefficient Payment Processing
  + Fees are computed manually, making the process prone to mistakes and delays, especially when multiple customers arrive at the same time.
* Unreliable Entry and Exit Tracking
  + Since there is no system-generated identifier, such as a QR code, for vehicles, it is difficult to accurately log parking duration and prevent disputes about time-in and time-out.
* No User Security
  + Anyone can access and update logbooks without leaving a trace, making it difficult to monitor staff actions and ensure accountability.

**Objectives of the Study**

The main objective of this study is to design and implement a Parking Management System for ParkZone Solutions that will replace its manual operations with a more efficient, accurate, and reliable system. Specifically, this study aims to:

* To develop an automated recording system that will accurately log vehicle entries, exits, and payments to reduce human error and missing records.
* To provide an organized slot allocation feature that will display real-time availability of parking spaces, preventing confusion, double allocation, and inefficient use of slots.
* To implement an automated payment processing system that will compute parking fees instantly, minimizing delays and ensuring accurate transactions during peak hours.
* To introduce a QR-based entry and exit tracking mechanism to monitor parking duration accurately, streamline vehicle logging, and prevent disputes on time-in and time-out.
* To establish secure staff authentication and accountability measures by assigning individual accounts, ensuring that all user activities are traceable and well-monitored.

**Review of Related Literature**

Local

Cruz (2020) conducted a study in Quezon City on the implementation of a computerized parking system to address the increasing number of vehicles in urban areas. The research highlighted several issues in manual operations, such as errors in recording and delays in transaction processing. This supports the objective of the present study, which aims to automate ParkZone Solutions’ parking management system for improved reliability.

Santos and Dela Cruz (2021) explored a parking monitoring system for universities in Manila. Their study focused on vehicle monitoring technologies and the challenges caused by congestion in school environments. The researchers found that automated parking systems enhance safety, speed up vehicle access, and reduce human error. This is aligned with our study’s goal of utilizing automation to reduce congestion and improve transaction speed at ParkZone Solutions.

Reyes (2022) analyzed parking systems in private malls in Makati, identifying common issues experienced by customers during peak hours such as long queues, slow ticketing, and inefficient slot allocation. The study found that shifting to digital systems leads to faster transactions, more accurate slot tracking, and higher customer satisfaction. Similarly, our research emphasizes the importance of a fast, reliable, and customer-friendly parking service.

Villamor (2022) examined UI/UX design preferences of Filipino users in mobile applications and found that usability, simplicity, and readability greatly influence user experience. Similarly, Pangilinan (2021) emphasized that consistent navigation and clear interface design improve accessibility in academic systems. These findings support the development of a user-friendly interface for the proposed Parking Management System.

Fernandez (2023) conducted a study in Mandaluyong City focusing on digital ticketing and QR-based verification in small commercial parking areas. The study showed that QR tickets reduced verification time and minimized disputes related to manual logging. These results support the use of QR technology in modernizing local parking operations, aligning with the present study’s approach to automated entry and exit validation.

Foreign

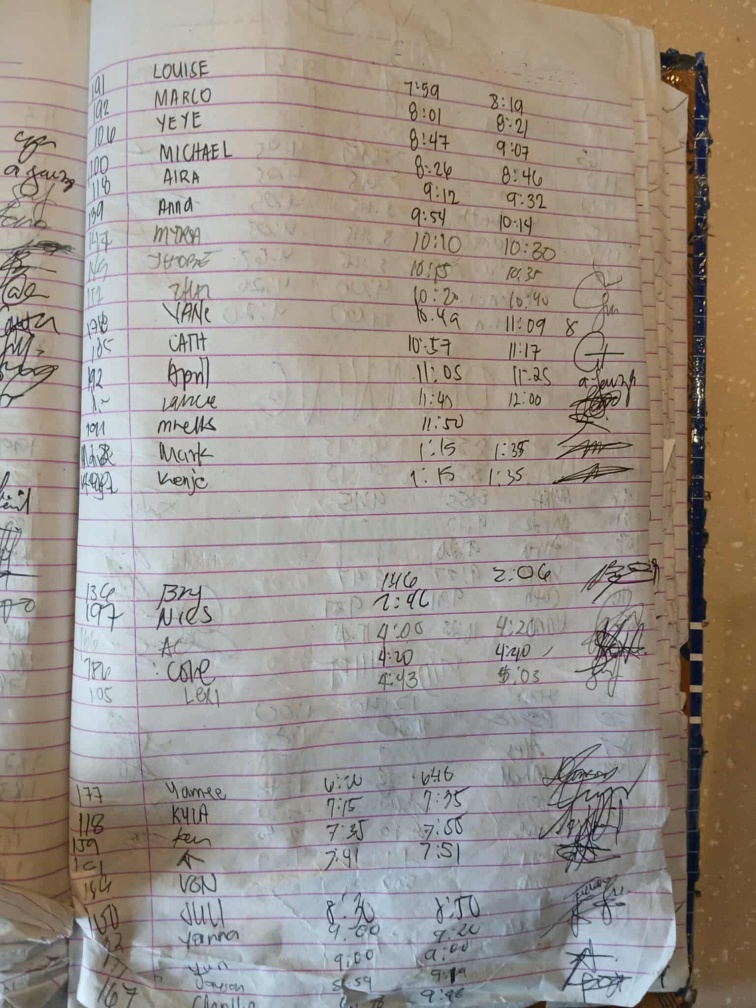
Lim et al. (2020) in Malaysia developed a smart parking system using sensors to detect vehicle occupancy. Their findings revealed that real-time data enhances parking flow, reduces congestion, and increases the overall efficiency of parking facilities. While their system uses sensors, the present study shares the same goal of improving operations through automation, using QR-based tracking instead.

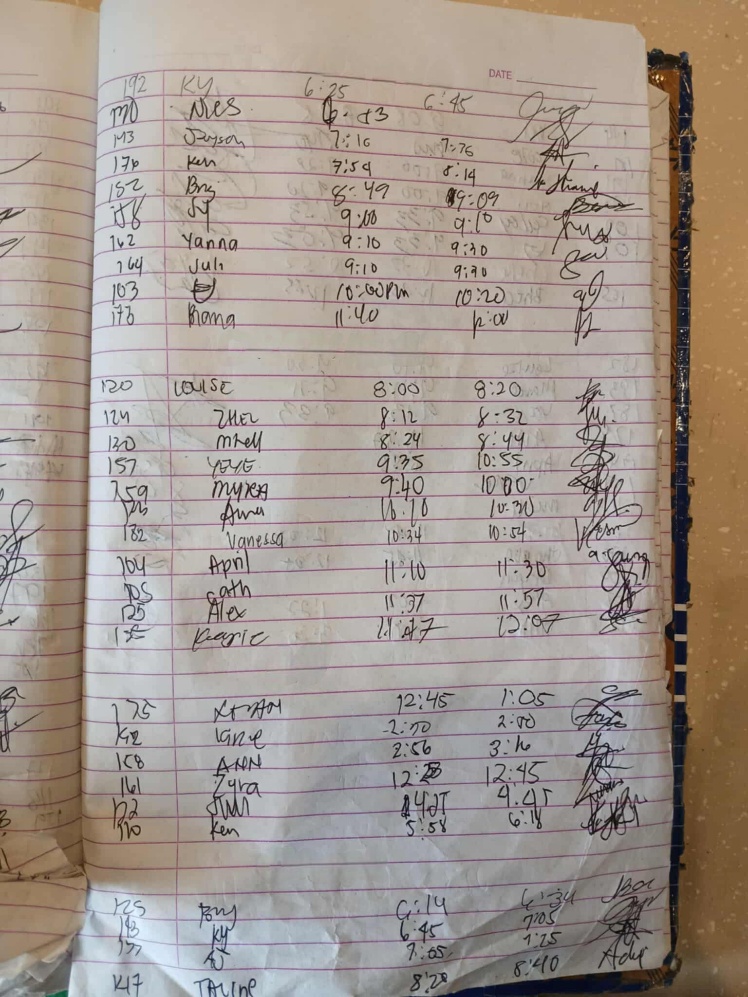
Sharma and Gupta (2020) in India examined QR-based automated parking systems. Their study found that QR technology improves transaction speed, minimizes fraud, enhances authentication, and ensures accurate time-in/time-out tracking. This is highly relevant to the current study since ParkZone Solutions will also implement QR codes for vehicle entry and exit.

Johnson (2021) in the United States explored cloud-based parking management systems and highlighted their benefits, including enhanced data security, improved report generation, and efficient remote monitoring. Similarly, the proposed system aims to leverage database-driven automation to improve operational efficiency.

Beyond parking systems, usability studies also support the interface design of the proposed system. Nielsen (2022) emphasized the importance of heuristic principles and minimalist design in improving user interaction and efficiency. Huang and Yen (2020) found that harmonious color schemes enhance user trust and comfort, while Michailidou (2021) demonstrated that simpler layouts improve visual processing and satisfaction. These studies reinforce the decision to combine automation with an intuitive and visually clear interface for ParkZone Solutions.

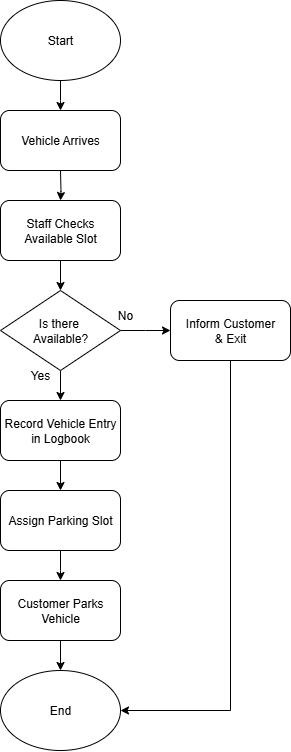
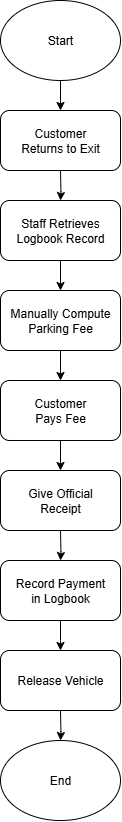
Ahmed and Rahman (2022) in Singapore developed an automated parking solution integrating QR codes with mobile payment systems. Their findings showed that QR-enabled transactions significantly reduced queues at entry points and improved accuracy in billing. This supports the current study’s adoption of QR technology to streamline the overall parking process at ParkZone Solutions.

**Old System Screenshot**

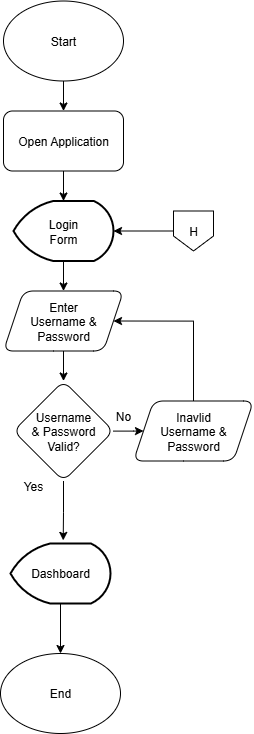




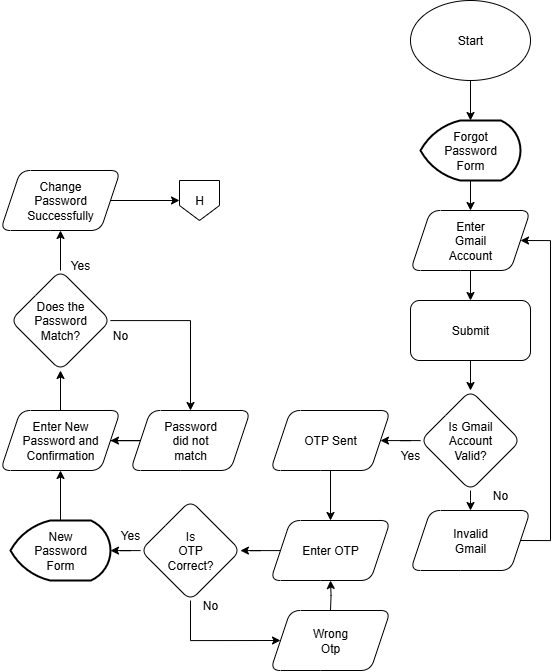
**Old System Flowchart**

****Parking entry process Parking exit process

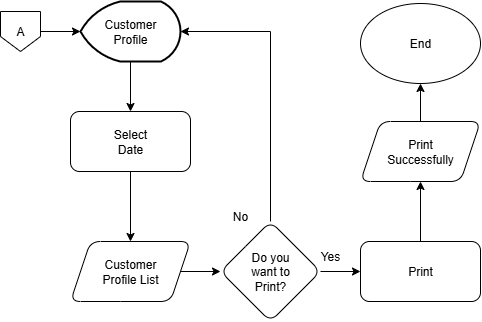
**Proposed System Flowchart**

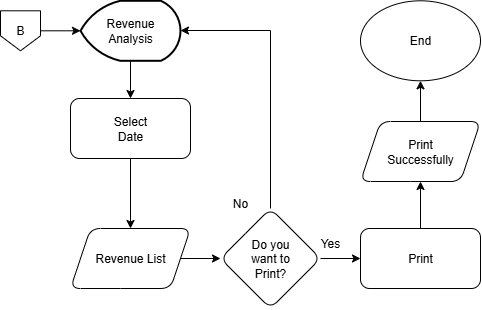
****Login

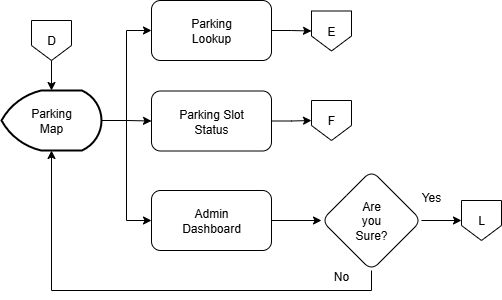
Forgot Password

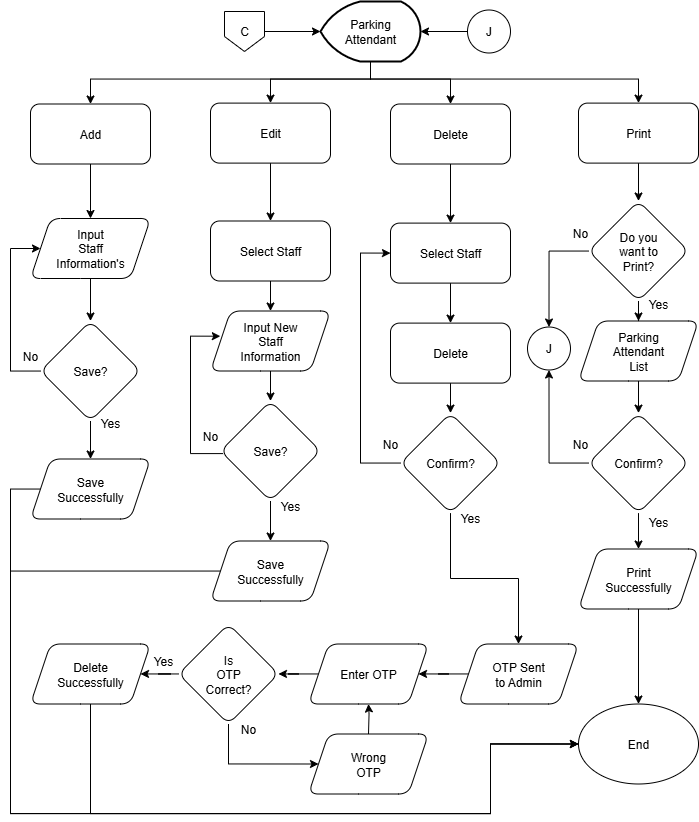
****

****Admin Dashboard

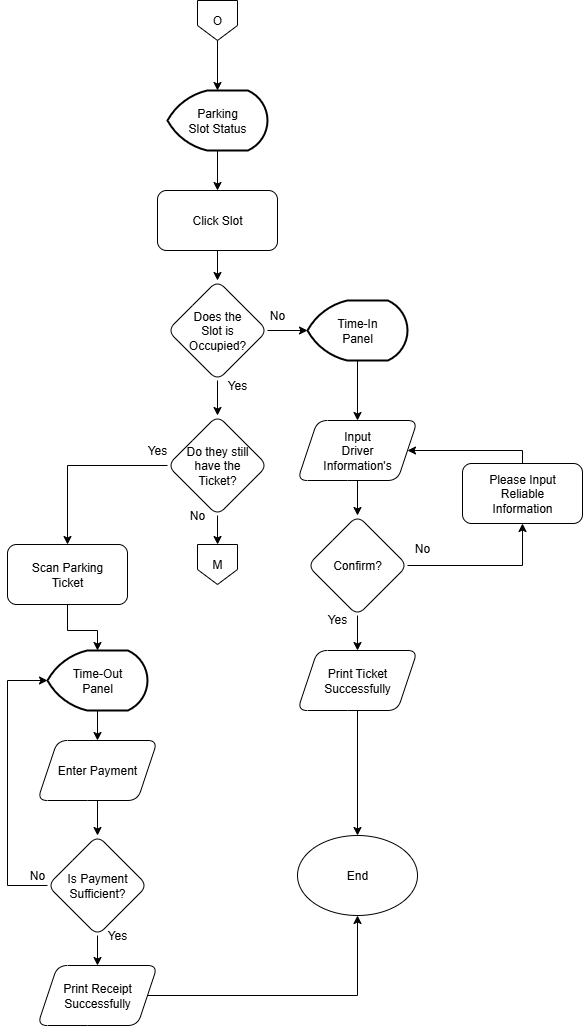
****Customer Profile Admin

****Revenue Admin

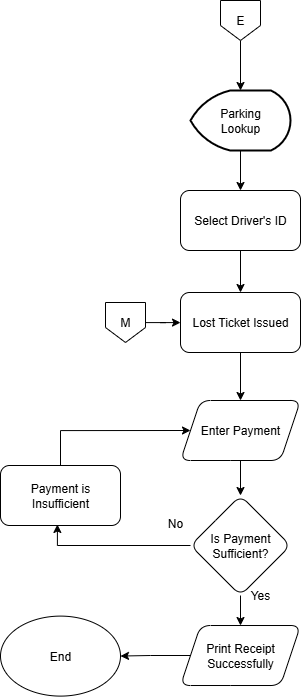
****Parking Map Dashboard Admin

****Parking Attendant Account Admin

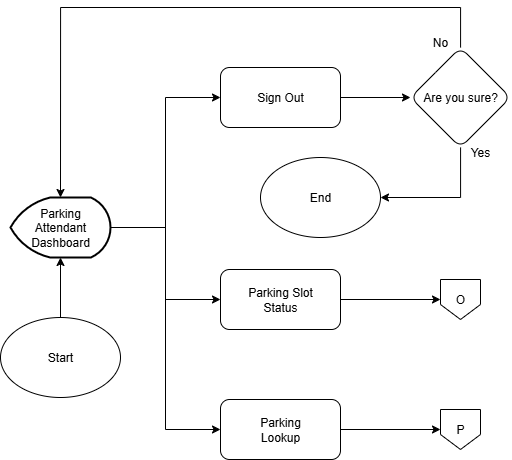
Parking Slot Status Admin

****

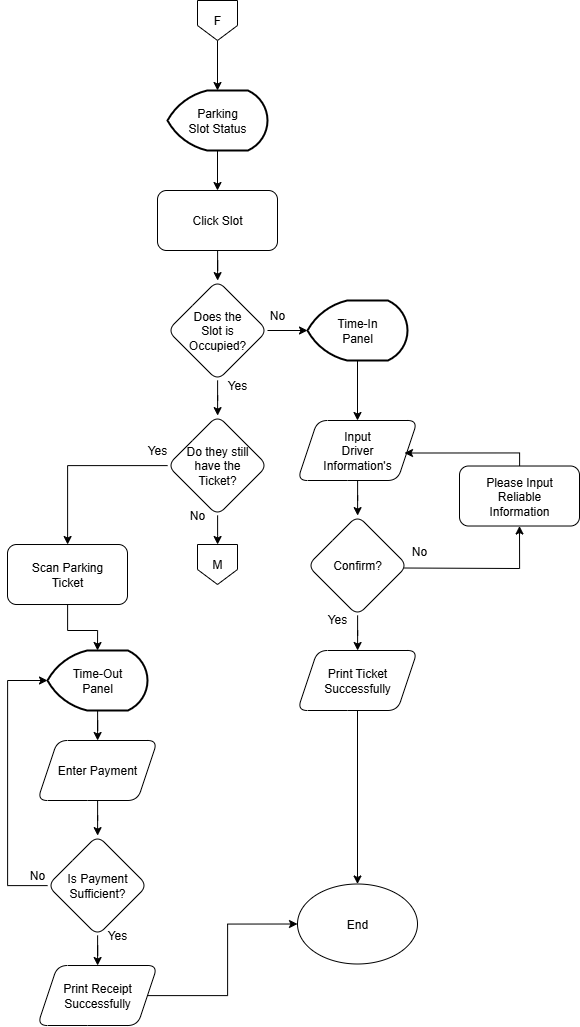
Parking Lookup Admin

****

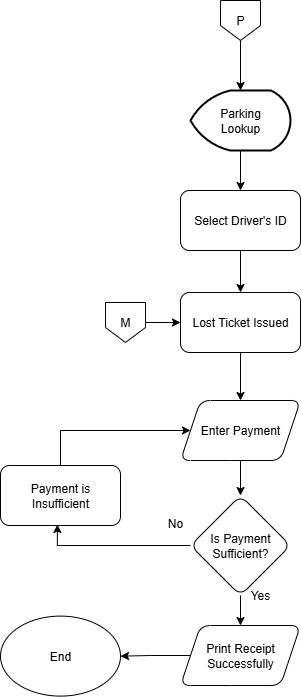
Parking Attendant / User Dashboard

****

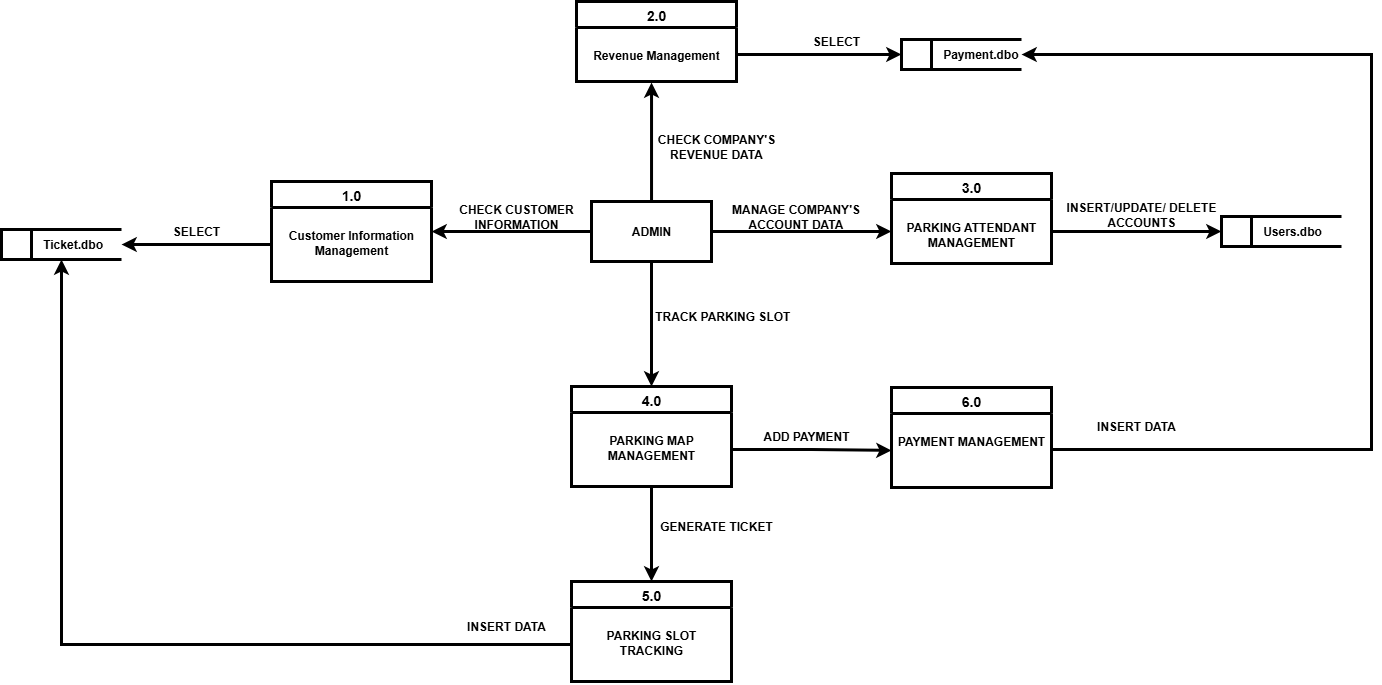
Parking Slot Status User

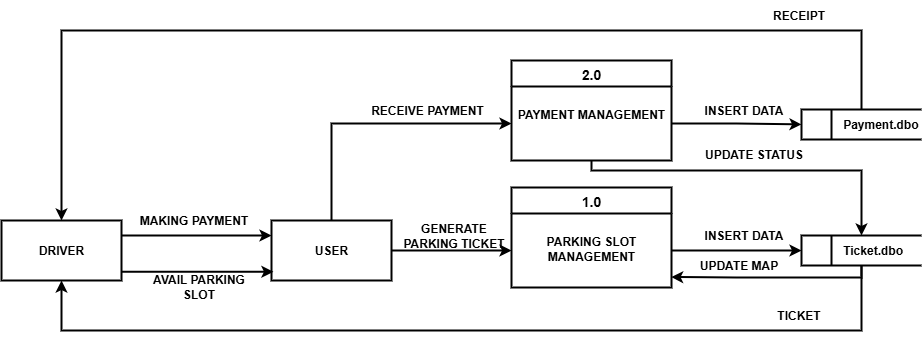
****

Parking Lookup User

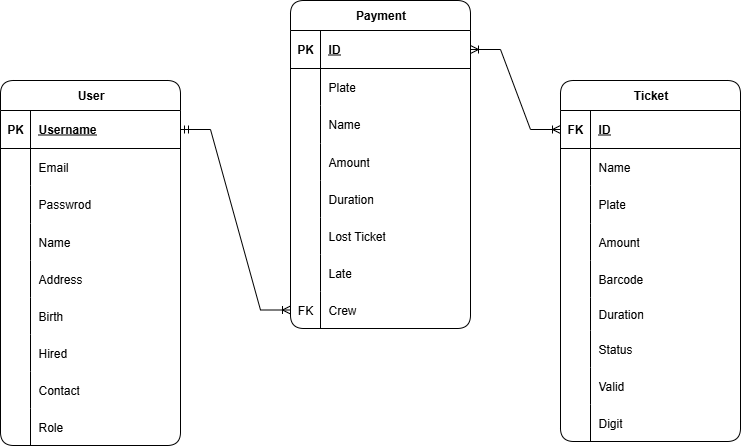
****

**Data Flow Diagram**

****Admin

****User

**Entity Relationships Diagram**

****

**Definition of Terms**

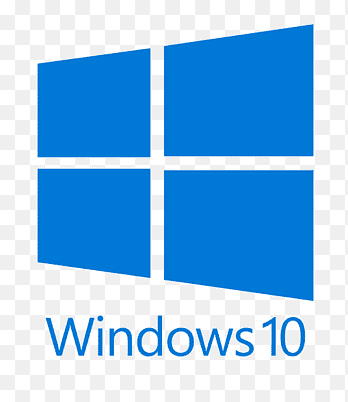
The following terminologies will be helpful in understanding the process of this research.:

* Automation – Refers to the process of using technology to perform tasks with minimal human intervention, such as recording entries and exits automatically.
* Database – A structured collection of data where vehicle, customer, and payment records are stored securely.
* Interface – The user-friendly screen or dashboard where staff can interact with the system.
* Monitoring – The continuous checking and tracking of parking slots and vehicle status within the system.
* Transaction – The complete process of payment for parking services recorded in the system.
* Accountability – The responsibility assigned to staff since their activities are logged and traceable in the system.
* Reporting – The automatic generation of daily, weekly, or monthly summaries of parking data.
* Tracking – The recording and following of vehicle movements such as entry and exit within the parking facility.
* Computation – The automatic calculation of parking fees by the system.
* Security – The protection of data and operations within the system through logins and controlled access.

**System Requirements**

Minimum

* Processor: Intel Core i3 or equivalent (CPU)
* Memory (RAM): 4GB
* Storage: 128 GB HDD or higher



* Operating System: Windows 10 (64-bit)

Recommended / Maximum

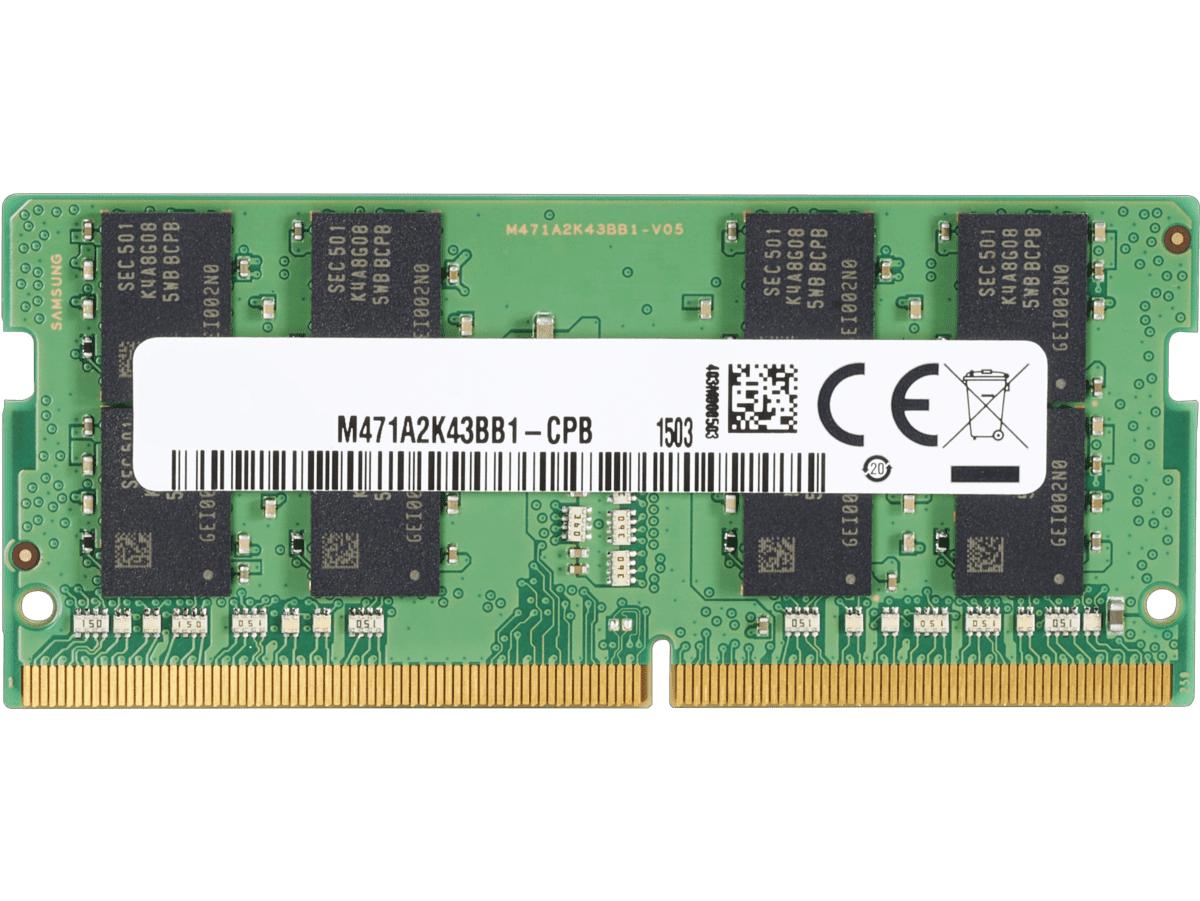
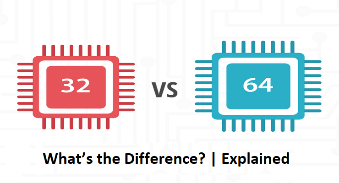
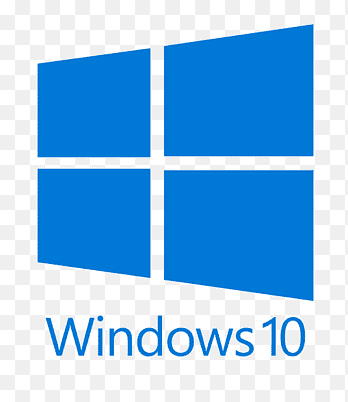
* Processor: Intel Core i7 or equivalent (CPU)
* Memory (RAM): 8GB or higher RAM
* Storage: 512 GB SSD or higher



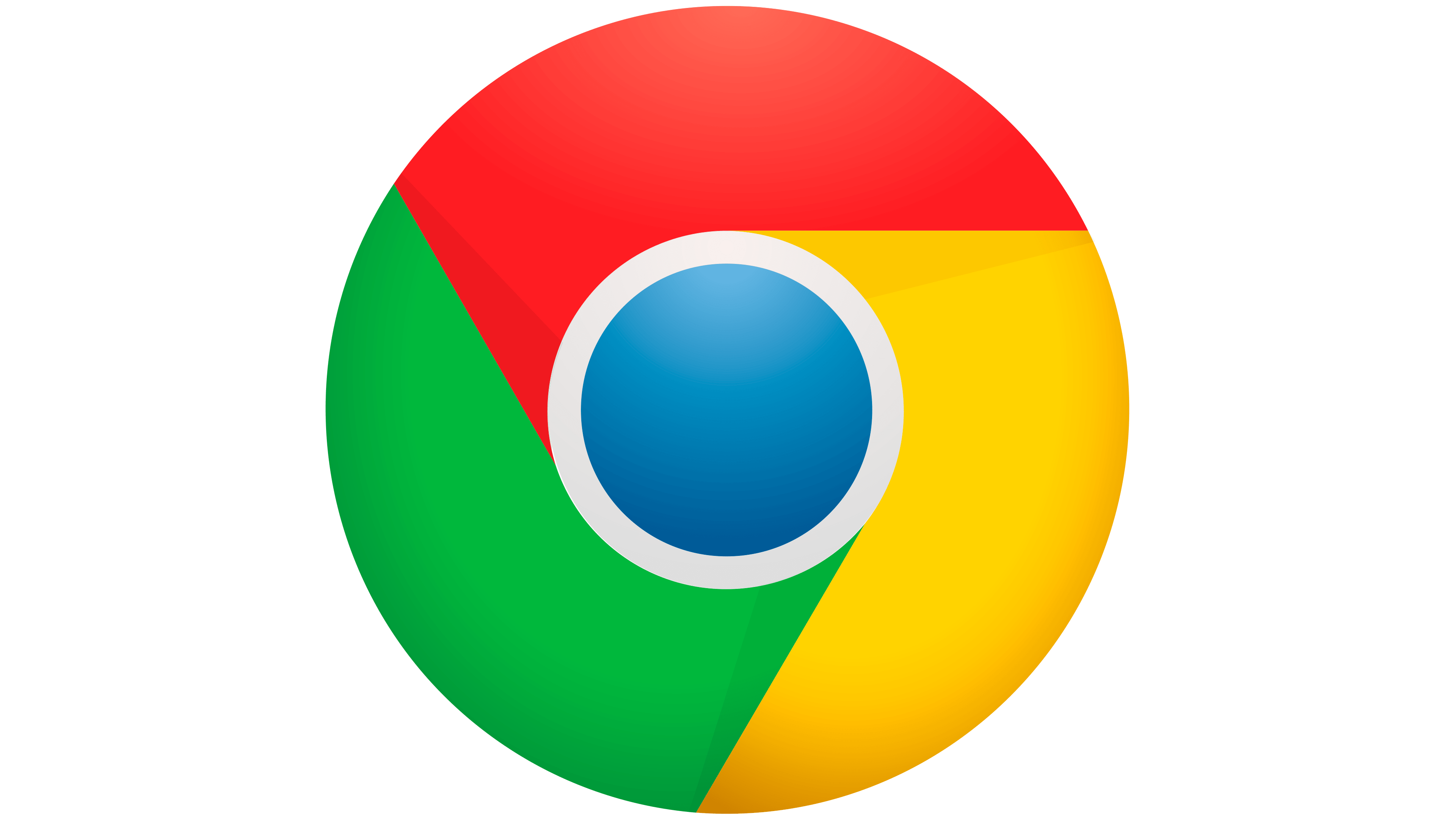
* Operating System: Windows 11 (64-bit)

**Software and Hardware Used**

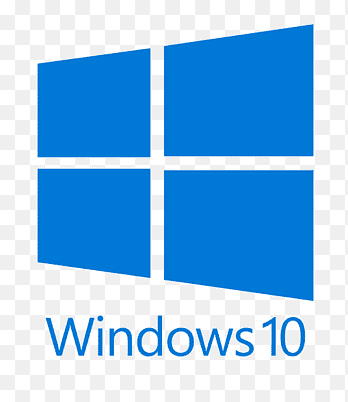
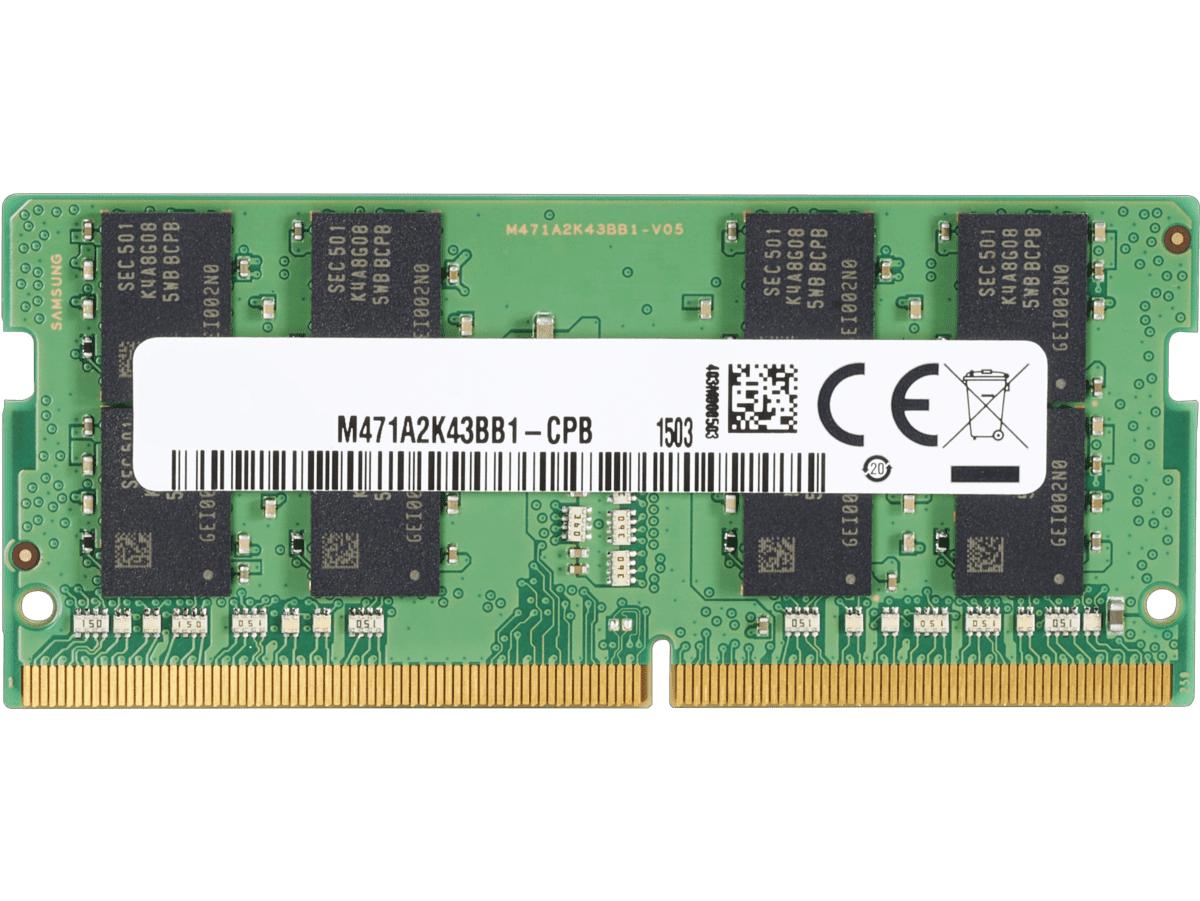
Hardware

* HP ProBook
* Intel Core i5-10210U (1.6 GHz base, up to 4.2 GHz)
* 8 GB RAM DDR4
* 64-bit operating system, x64-based processor
* Windows 10 Pro

Software

* Visual Studio
* Draw IO
* Microsoft Word
* SQL Server Management Studio 2025
* Net Framework 4.8
* Google Chrome
* Opera Browser

Computer Specification Tested

* DESKTOP-49ID4OC
* Intel Core i3 7th Gen
* Microsoft Windows 10 Pro
* 4 GB RAM
* 512 GB HDD Storage

**Cost and Benefits**

|  |  |  |
| --- | --- | --- |
| **Item** | **Description** | **Estimated Cost (PHP)** |
| System | Full Development of the Parking Manangement System | ₱7,000 |
| Maintenance | System Updates, Fix Bugs and Technical Support | ₱2,000 (per month) |
| Hardware Option 1 | Laptop | ₱30,000 |
| Hardware Option 2 | Computer Set with Web Cam / Camera | ₱15,000 |
| Printer | Thermal Receipt Printer and Standard Inkject Printer | ₱7,000 |
| **Total** | **Hardware Option 1**  **₱46,000** | **Hardware Option 2**  **₱31,000** |

Cost

Benefits

The proposed Parking Management System significantly improves the efficiency and accuracy of ParkZone’s Solutions operations by automating the recording of vehicle entries, exits, and payments, reducing errors caused by manual encoding. Its real-time slot monitoring ensures organized parking allocation and prevents double booking, while the automated fee computation speeds up transactions during peak hours. The QR-based entry and exit tracking provides faster and more accurate monitoring of parking duration, minimizing disputes. Additionally, secure staff authentication strengthens accountability by ensuring that all system activities are properly tracked and authorized.

**Summary**

This study presents the development of a Parking Management System designed to address the operational challenges of ParkZone Solutions, located at 781 M.V. Delos Santos St., Sampaloc, Manila. The existing manual processes, such as handwritten logbooks, visual slot checking, and basic spreadsheet tracking, have led to inaccuracies, delays, and a lack of staff accountability. To overcome these issues, the proposed system automates core operations, including vehicle entry and exit logging, slot management, fee computation, payment recording, and report generation. It also integrates a secure login feature to ensure accountability among staff, standardizes the ₱50 per hour rate, and introduces an accurate late fee system to improve billing precision and fairness.

By implementing this automated solution, ParkZone Solutions can transition from a traditional manual process to a more efficient, transparent, and reliable digital system. The Parking Management System minimizes human error, enhances monitoring and record-keeping, speeds up transactions, and improves customer satisfaction. Overall, this project demonstrates how automation can transform a conventional parking facility into a modern, technology-driven service that meets the growing demand for convenience, consistency, and efficiency in urban parking management.

**Recommendations**

Based on the findings of this study, we highly recommend that ParkZone Solutions implement the proposed Parking Management System and adopt the solutions designed to resolve the issues in their current manual process. The system should automate vehicle entry and exit recording, as well as payment transactions, to ensure accuracy and reduce human errors. Organized slot allocation must also be implemented, allowing staff to monitor available parking spaces digitally rather than relying on manual observation. To further improve reliability and customer experience, we recommend using a QR-based entry and exit tracking feature, which will automatically record the exact time-in and time-out of each vehicle and prevent disputes regarding parking duration.

Additionally, we recommend the use of automated payment computation and report generation to achieve faster transactions and more efficient financial tracking. Each staff member should have their own personalized user account to ensure accountability, with all activities being traceable through the system’s secure login feature. By applying these automated solutions and standard operating procedures, ParkZone Solutions can modernize its daily operations, enhance efficiency, minimize human error, and provide customers with a faster, more reliable, and technology-driven parking experience.

**Data Gathering**

A group of people standing in front of a parked car

AI-generated content may be incorrect.A group of people standing in front of a car

AI-generated content may be incorrect.

A person and person pointing at something

AI-generated content may be incorrect.A group of people walking in a parking lot

AI-generated content may be incorrect.

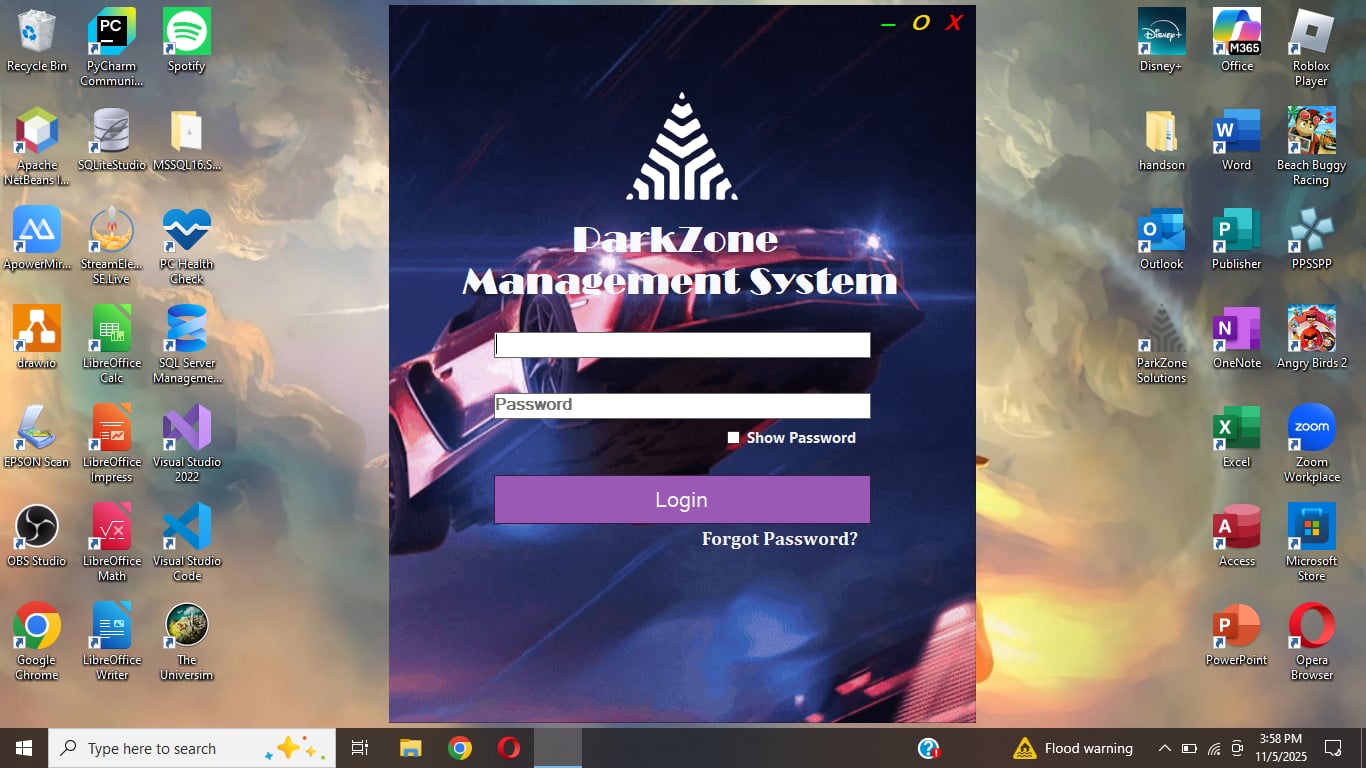
We conducted an in-person interview last 18th of August 2025 to identify and better understand the real problems within the company.

**Conclusion**

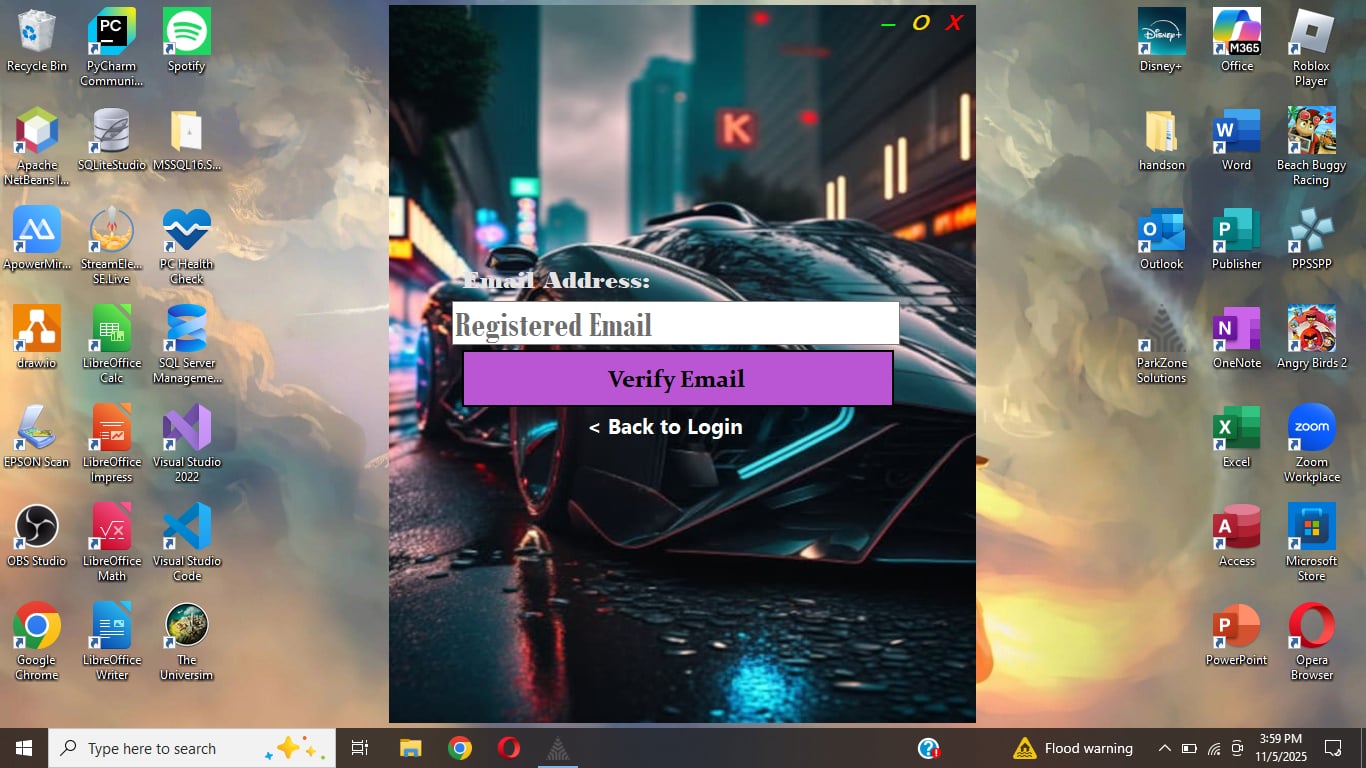
Therefore, we conclude that the current manual system of ParkZone Solutions is no longer sufficient to meet the demands of modern parking management. The reliance on handwritten logs, observation-based slot monitoring, and basic spreadsheets cause delays, inaccuracies, and inefficiencies that negatively affect both the staff and customers. Through our study, we were able to identify the core problems such as inaccurate records, slow transactions, poor slot allocation, reporting difficulties, and lack of staff accountability, all of which hinder the overall performance of the parking operations.

Therefore, we believe that implementing an Automated Parking Management System is the most effective solution to these challenges. This system will not only streamline vehicle tracking, fee computation, and slot monitoring but also ensure more accurate records, real-time reporting, and better accountability among staff. By adopting this system, ParkZone Solutions can modernize its operations, provide faster and more reliable services, and establish itself as an efficient and technology-driven parking facility in Metro Manila.

**New System Screenshot**

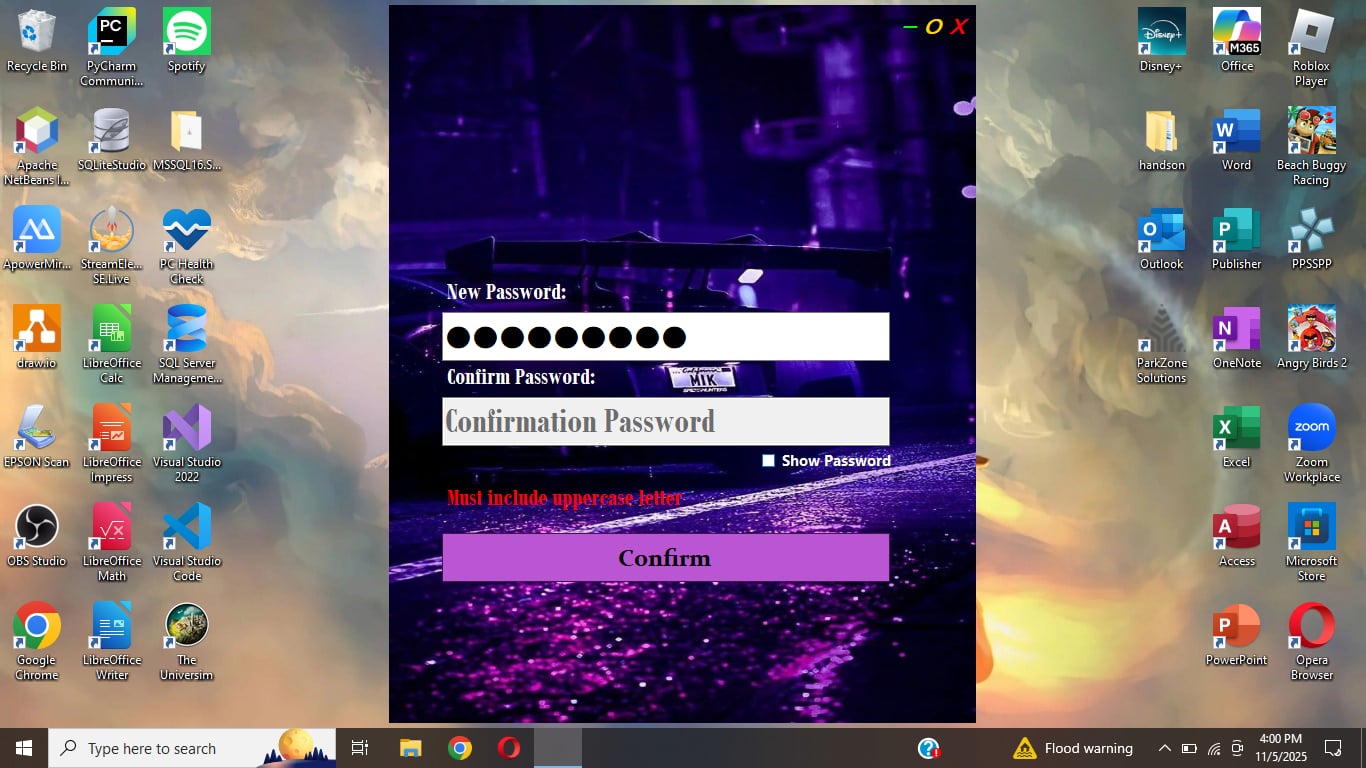
****Log In

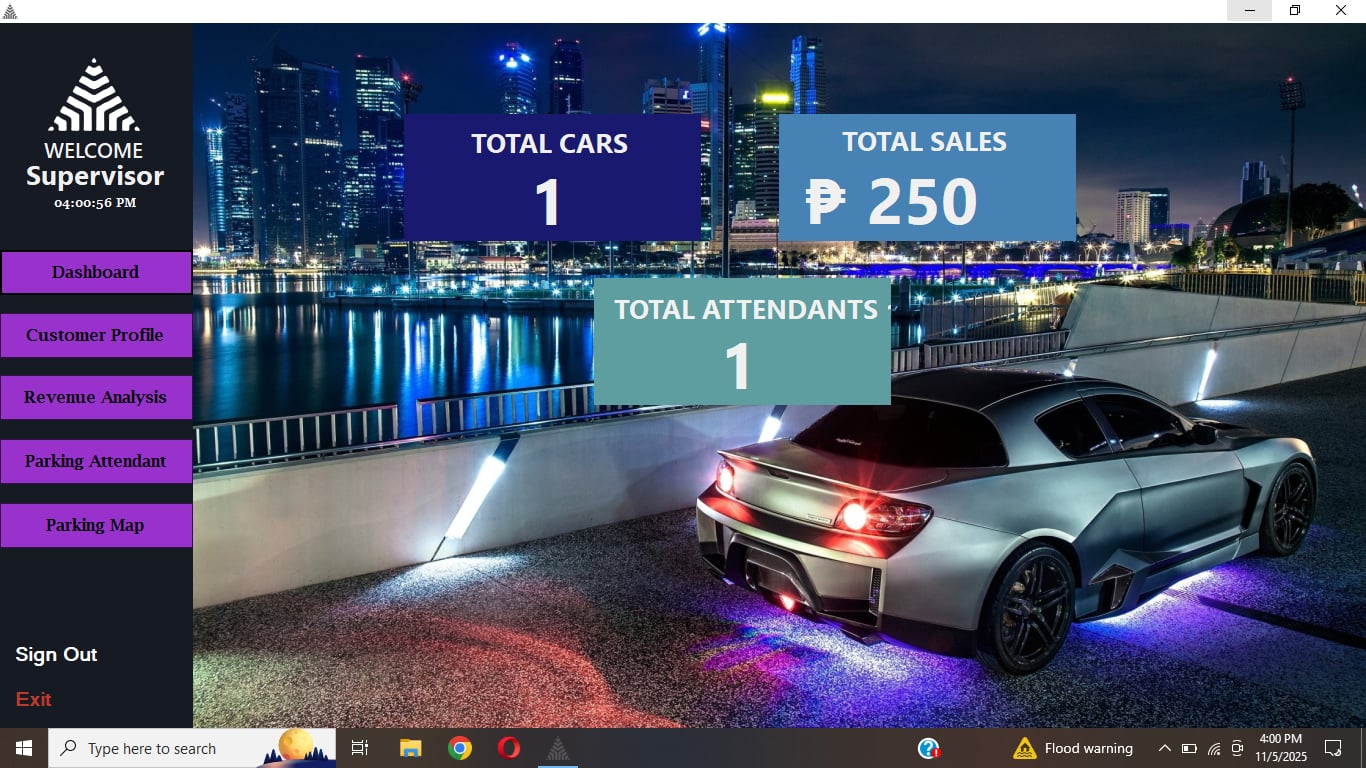
Email Confirmation

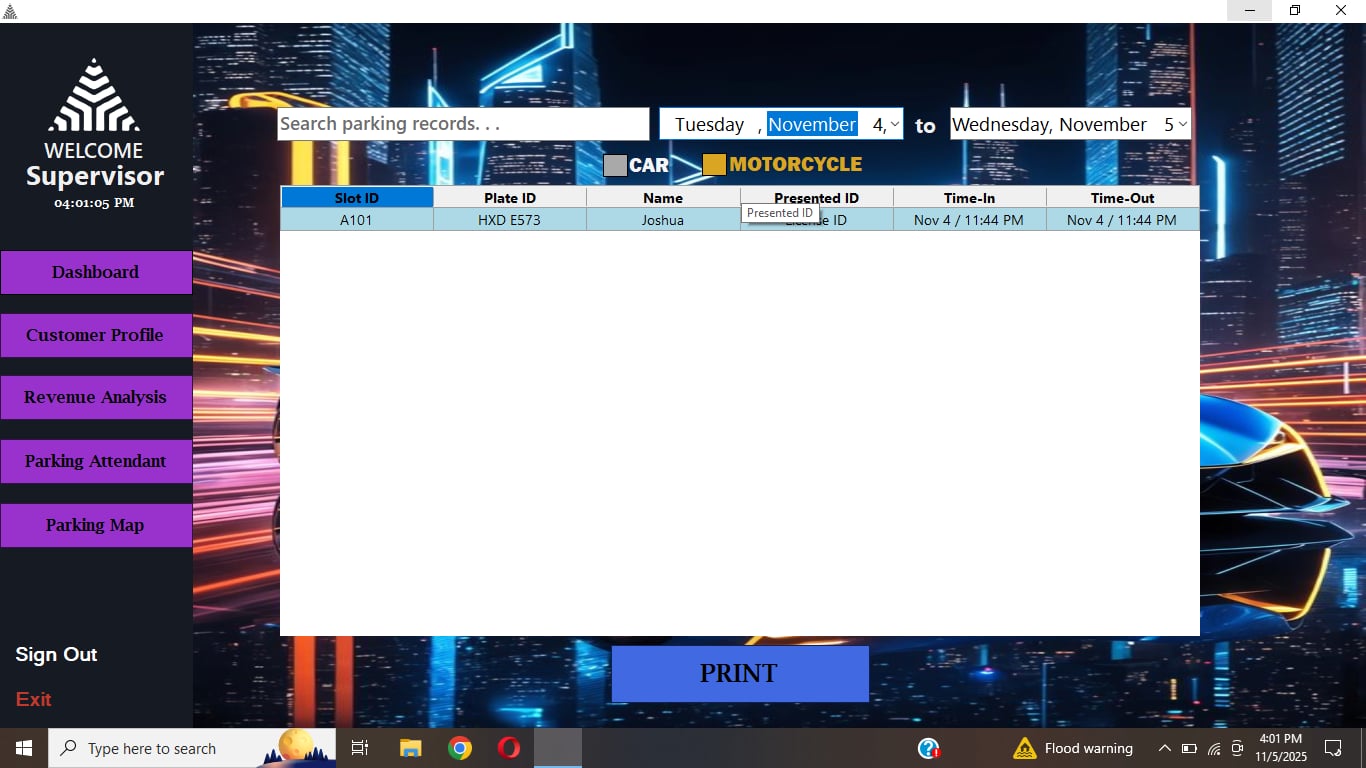
****

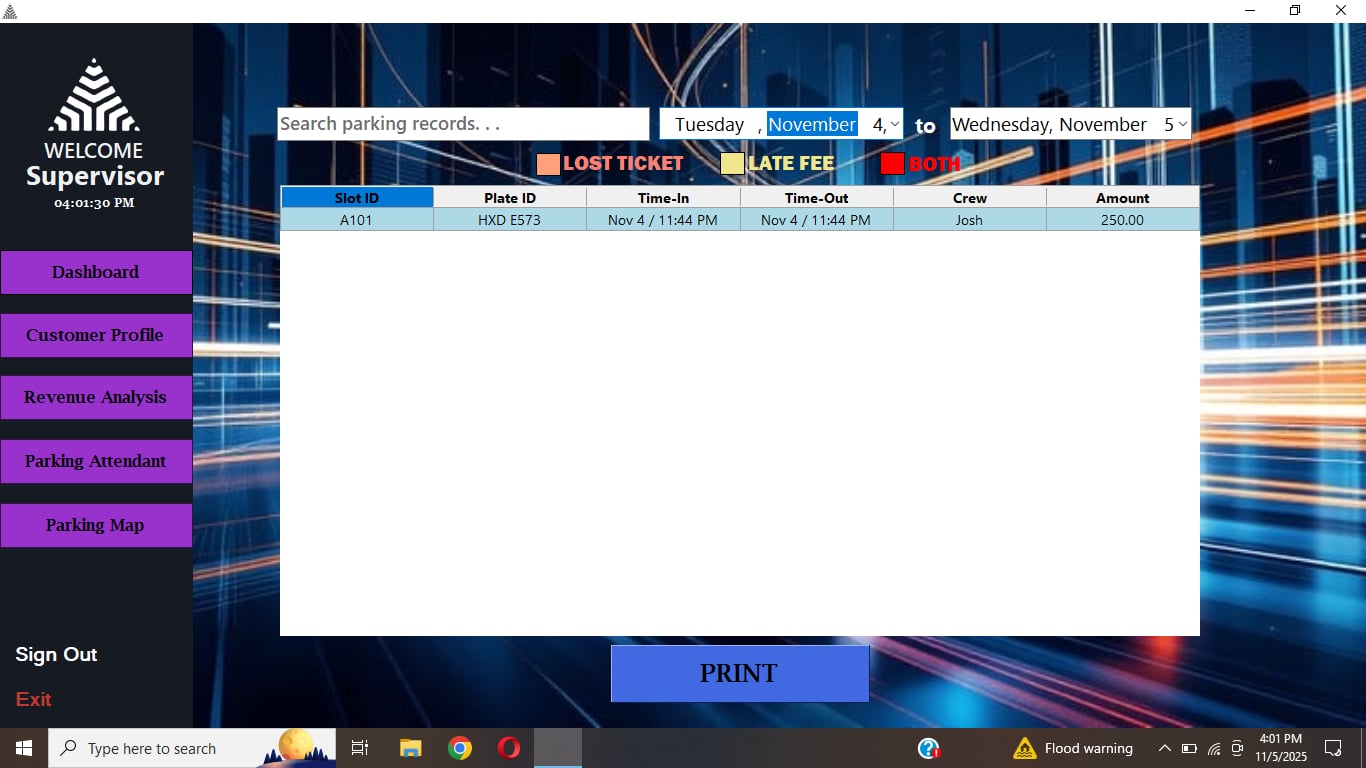
****

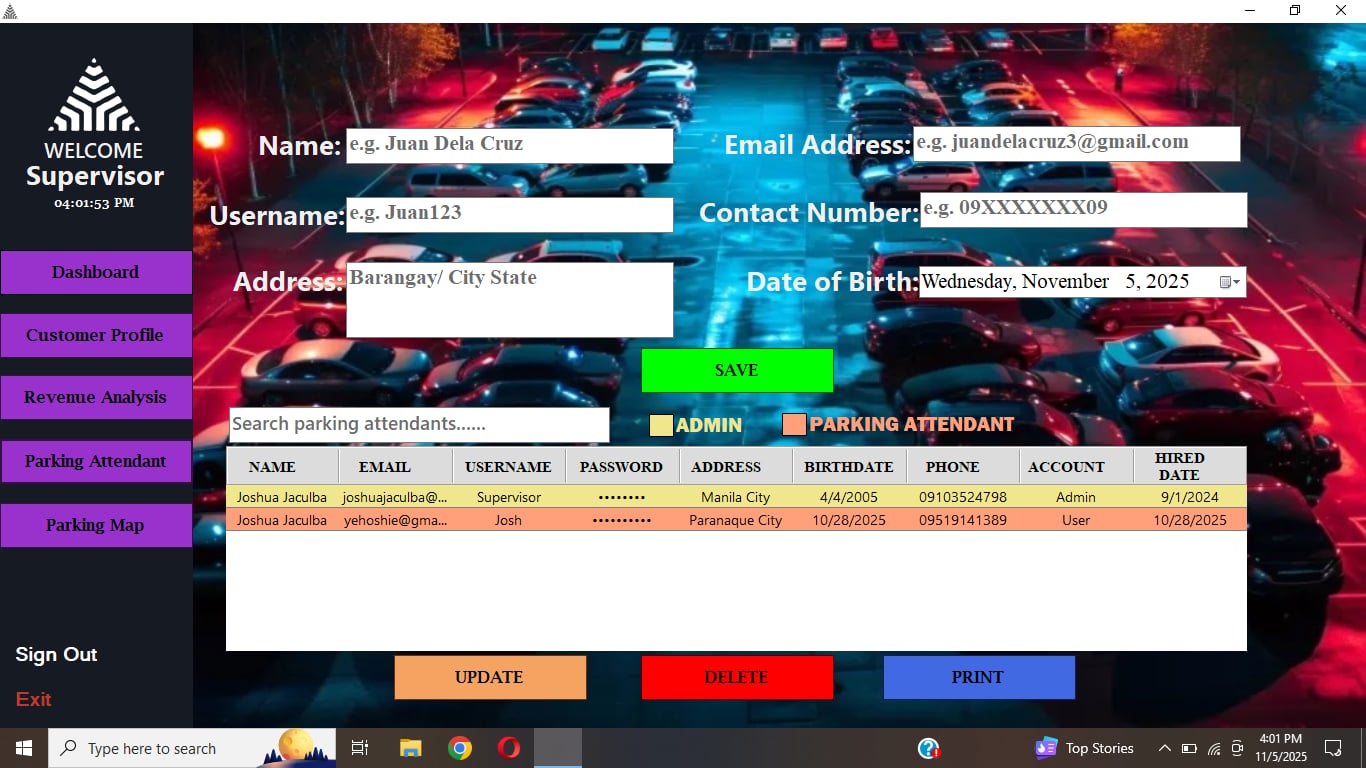
OTP Confirmation

****Change Password

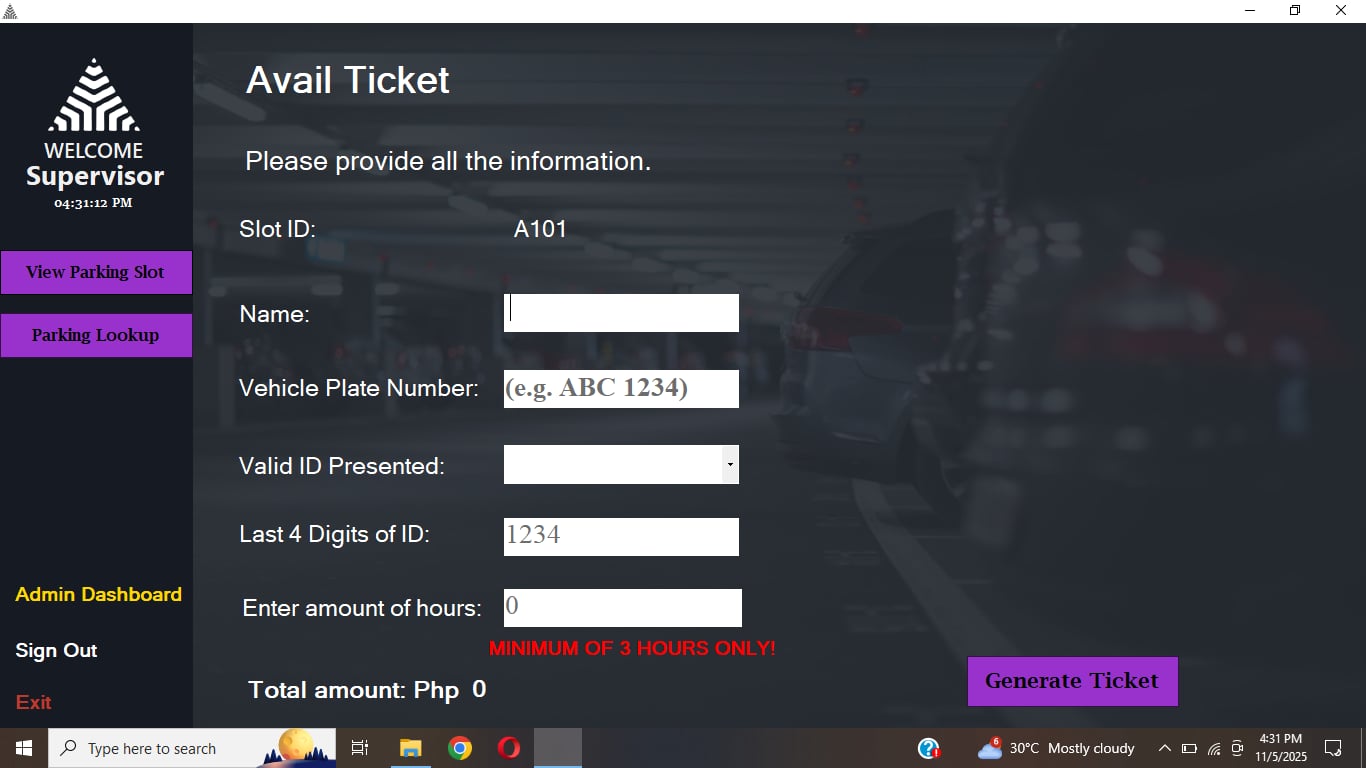
****Admin Dashboard

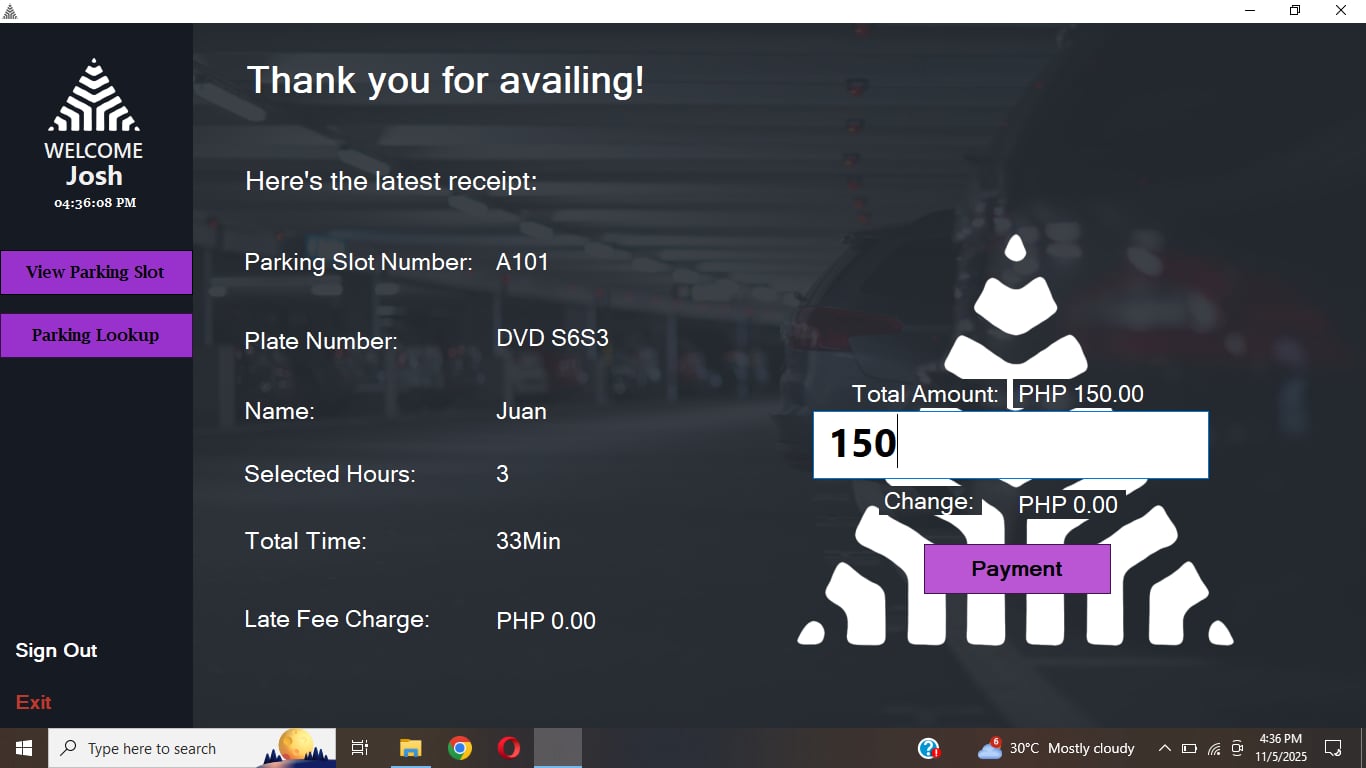
Customer Profile List

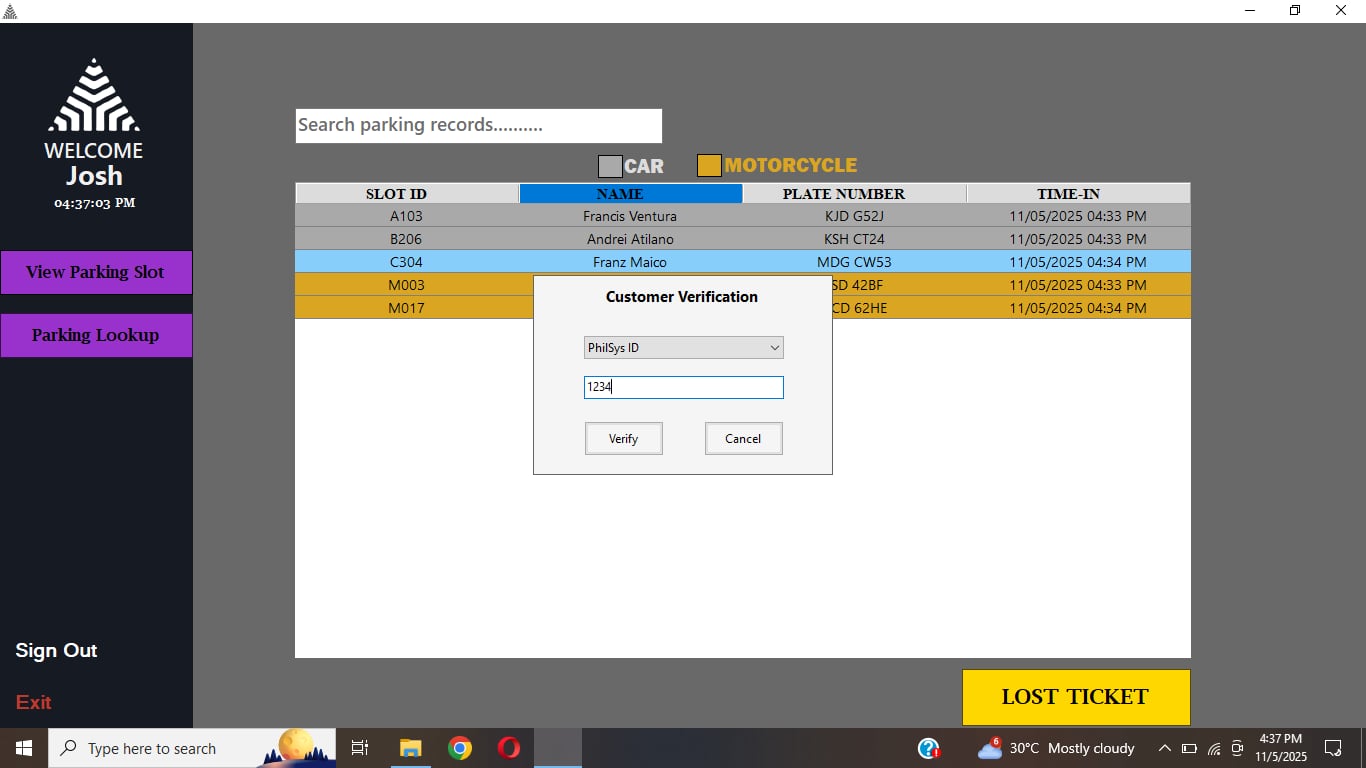
Revenue Analysis

User Account List

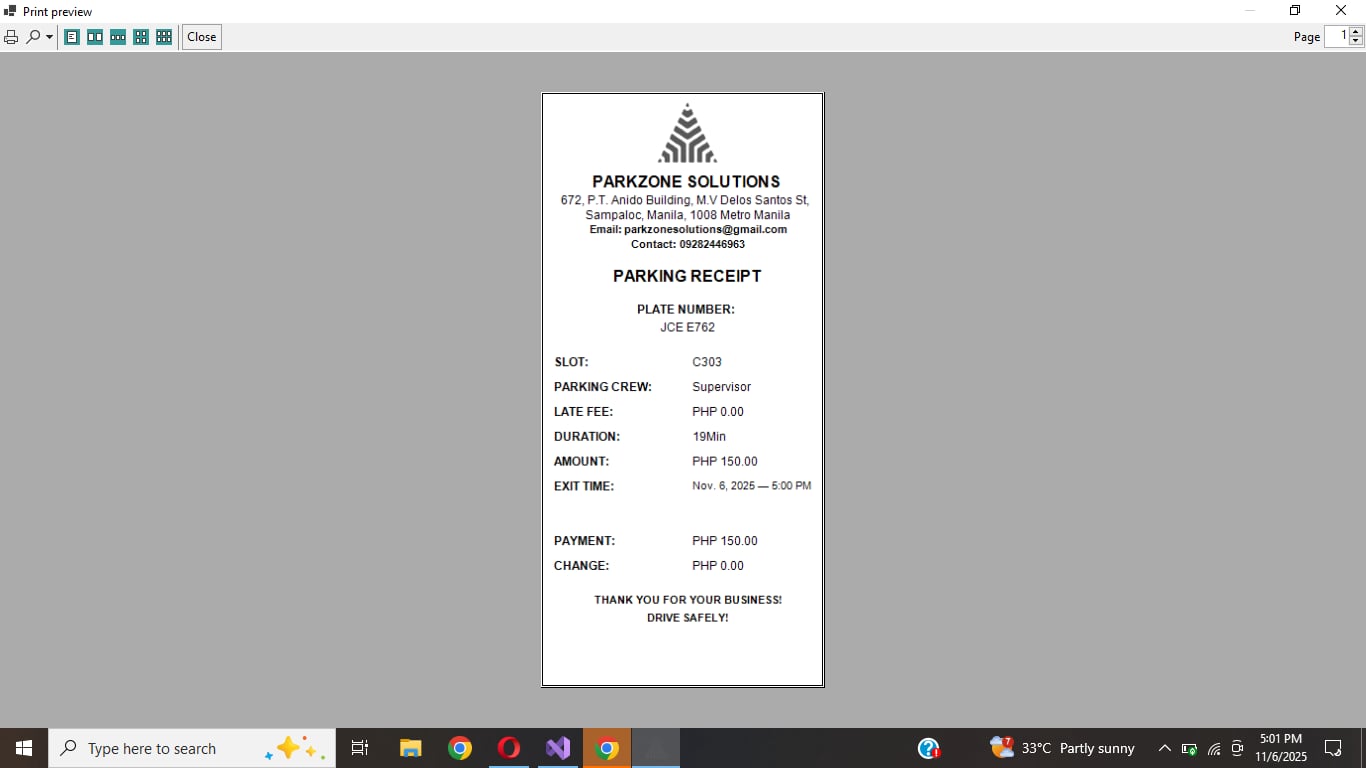
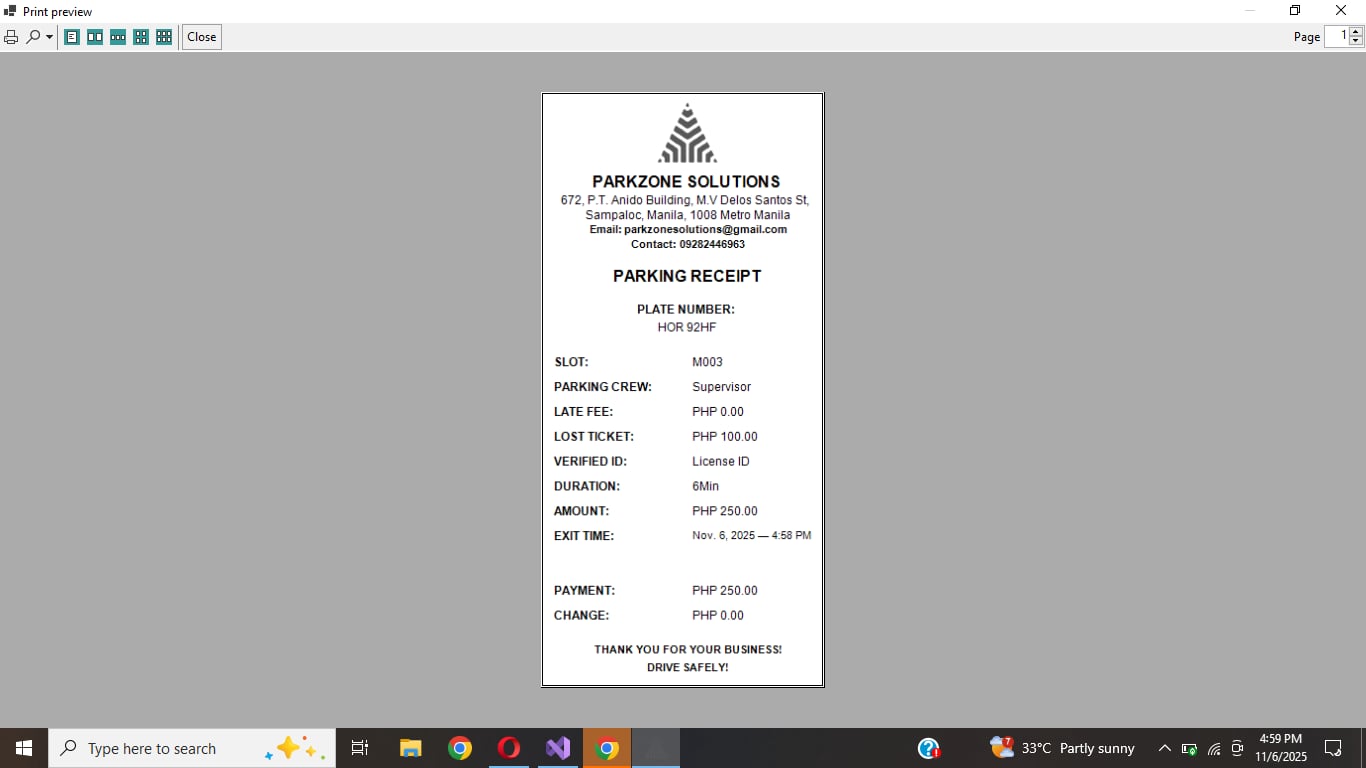
Parking Slot Availability

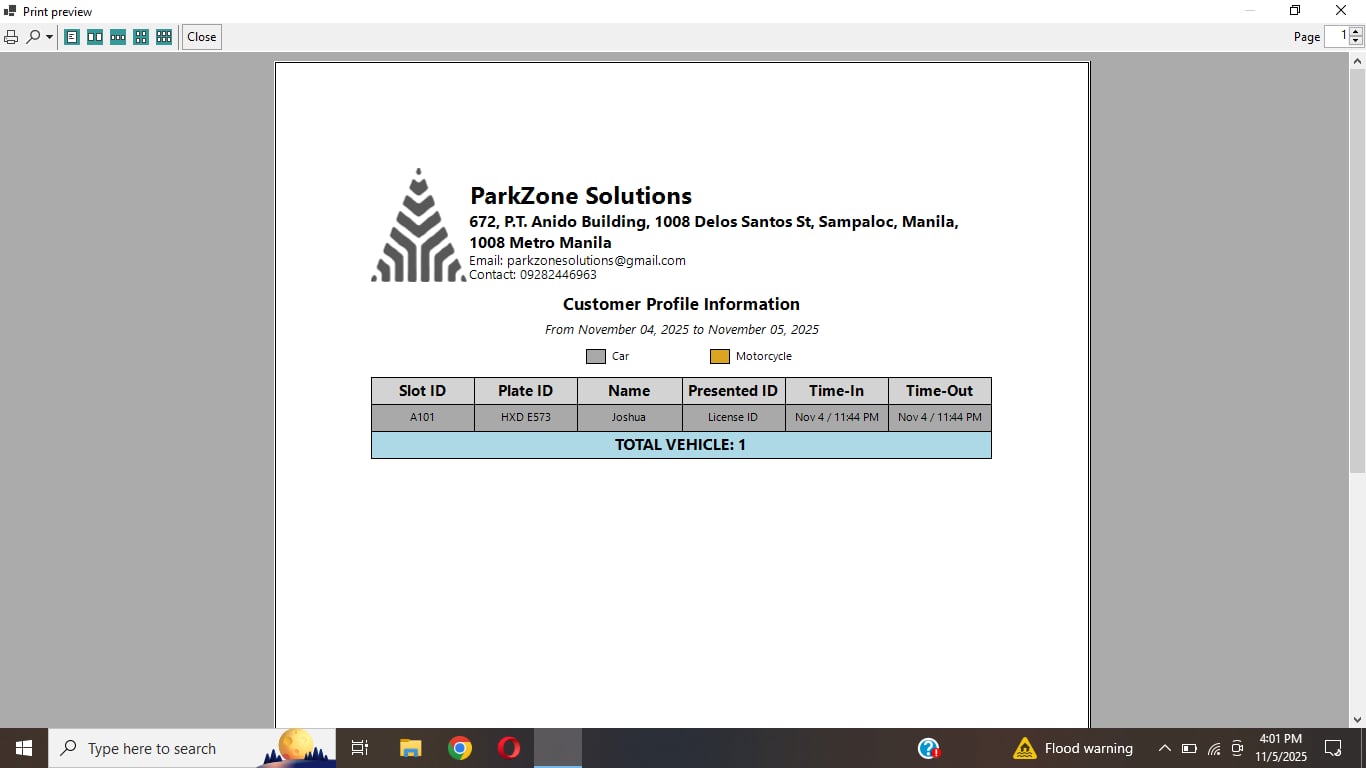
Ticket Issued

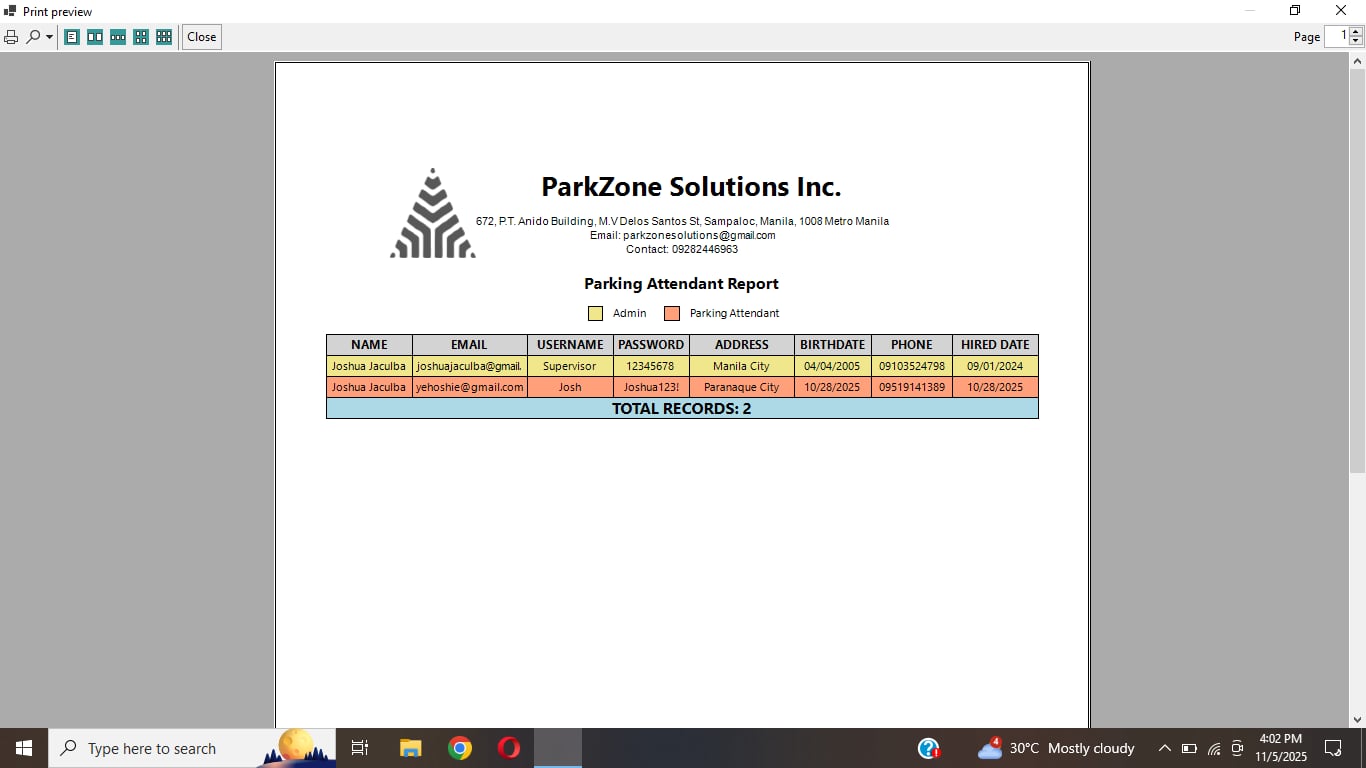
Billing

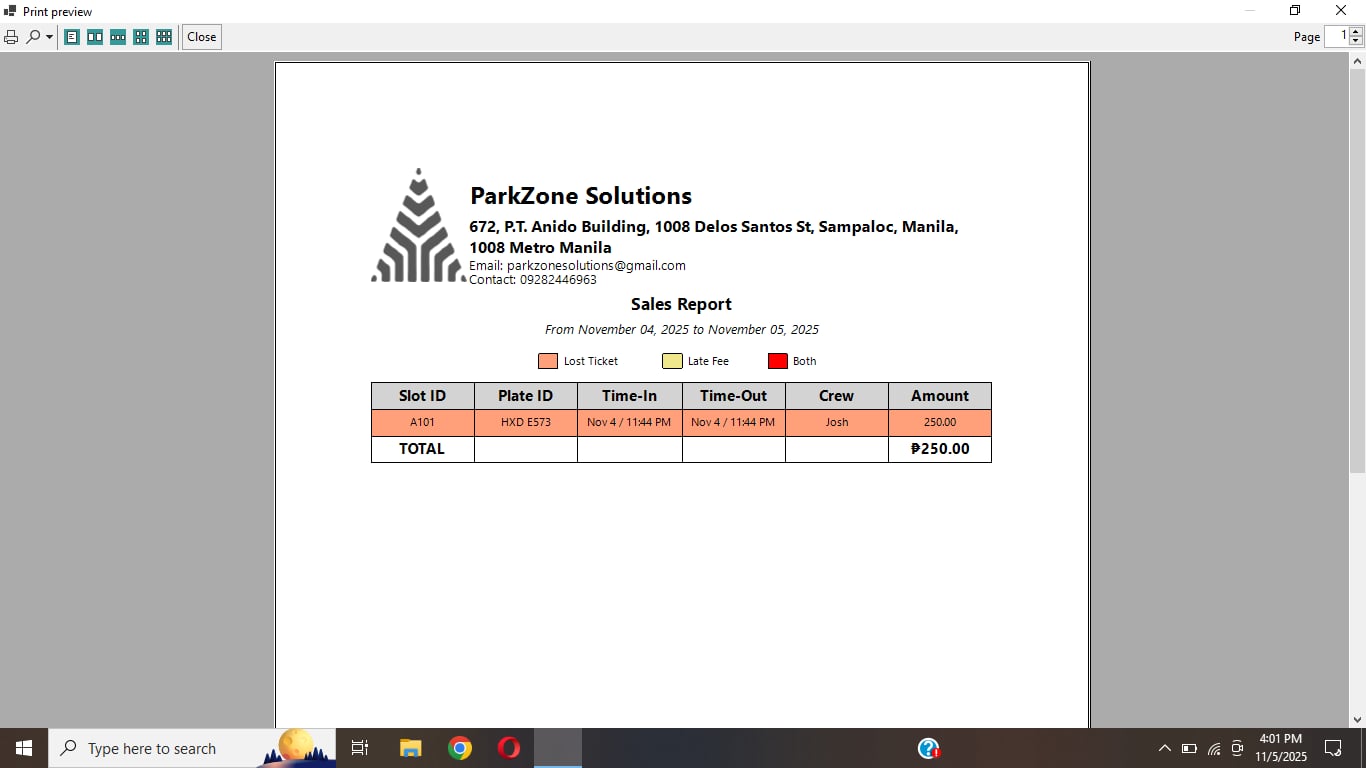
Lost Ticket Issued

**Sample Receipt**

****

****

****

****

**Installation Guide**

This guide outlines the step-by-step procedure for installing and running the Parking Management System.

STEP 1: Prepare the Device

1. Start up the PC or laptop.
2. Stable internet connection.

STEP 2: Download and Install Required Software

Download and install the following:

* Microsoft SQL Server Management Studio (SSMS)
* Microsoft® SQL Server® 2022 Express
* Project File (ZIP format)
* WinRAR (optional, for file extraction)

Step 3: Extract the Project File

1. Open File Explorer
2. Locate the downloaded ZIP file.
3. Right-click the file.
4. Select Extract or Extract Here (if using WinRAR).
5. Wait for the extraction to finish.

Step 4: Set Up the Database Connection

1. Open SQL Server Management Studio
2. Enter the server name provided by the SQL Server 2022 Express installation.
3. Check Trusted Server Certificate.
4. Click Connect.

Step 5: Import the Database

1. In SSMS, right-click Databases
2. Select Import Data-Tier Application.
3. Click Next.
4. Click Browse.
5. Locate Parking Management System.bacpac.
6. Click Next
7. Click Finish to complete the import

STEP 6: Install the Parking Management System Application

1. Open the extracted folder containing the System Installer.
2. Double-click the ParkingManagementSystemSetup.exe (or similar installer).
3. Follow the on-screen installation instructions:
   1. Click Next
   2. Choose the installation folder (or leave default)
   3. Click Install
4. Wait for the installation to complete.
5. Click Finish when prompted.

STEP 7: Launch the Application

1. Locate the newly installed Parking Management System in:
   1. Desktop shortcut (if created), or
   2. Start Menu → Parking Management System
2. Double-click to open the system.
3. Log in using the default credentials provided.
4. Test system features to verify database connectivity.

**Source Code**

**Log In**

Imports System.Data.SqlClient

Imports System.Drawing

Imports System.Reflection

Imports System.Text.RegularExpressions

Imports System.Windows.Forms

Public Class Login

Public username, uEmail

Private gif As Image

Private frameTimer As Timer

Private Sub Label3\_Click(sender As Object, e As EventArgs)

Dim file As StreamReader = My.Computer.FileSystem.OpenTextFileReader("userDB.txt")

Dim line As String

Dim userFound = False

While Not file.EndOfStream

line = file.ReadLine

Dim f = line.Split(", ")

If f(2) = email.Text AndAlso f(1) = password.Text Then

userFound = True

uEmail = f(2)

username = f(0)

Exit While

End If

End While

file.Close()

If userFound Then

userMainMenu.Panel1.Visible = False

userMainMenu.Show()

Hide()

End If

password.Clear()

End Sub

Private Sub Label4\_Click(sender As Object, e As EventArgs) Handles Label4.Click

Application.Exit()

End Sub

Private Sub Label2\_Click(sender As Object, e As EventArgs) Handles Label2.Click

Address.Show()

Address.Button1.Focus()

Hide()

End Sub

Private Sub email\_TextChanged(sender As Object, e As EventArgs) Handles email.TextChanged

Dim txt As String = email.Text

Dim caret As Integer = email.SelectionStart

txt = txt.Replace(" ", "")

txt = Regex.Replace(txt, "[^a-zA-Z0-9@.]", "")

Dim atCount = txt.Count(Function(c) c = "@"c)

If atCount > 1 Then

Dim i = txt.IndexOf("@")

txt = txt.Substring(0, i + 1) & txt.Substring(i + 1).Replace("@", "")

End If

email.Text = txt

If caret > txt.Length Then caret = txt.Length

email.SelectionStart = caret

End Sub

Private Sub email\_KeyPress(sender As Object, e As KeyPressEventArgs) Handles email.KeyPress

If Char.IsControl(e.KeyChar) Then Return

If Char.IsLetterOrDigit(e.KeyChar) Then Return

If e.KeyChar = "@"c Or e.KeyChar = "."c Then Return

e.Handled = True

End Sub

Private Sub email\_KeyDown(sender As Object, e As KeyEventArgs) Handles email.KeyDown

If e.KeyCode = Keys.Enter Then

e.SuppressKeyPress = True

Button1.PerformClick()

End If

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Dim connString As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Dim query As String =

"SELECT [Email], [passwrod], [Role]

FROM [dbo].[User]

WHERE [Email] COLLATE SQL\_Latin1\_General\_CP1\_CS\_AS = @Email

AND [passwrod] COLLATE SQL\_Latin1\_General\_CP1\_CS\_AS = @Password"

Using conn As New SqlConnection(connString)

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Email", email.Text.Trim())

cmd.Parameters.AddWithValue("@Password", password.Text.Trim())

conn.Open()

Dim r = cmd.ExecuteReader()

If r.Read() Then

Dim role As String = r("Role").ToString().ToLower()

If role = "user" Then

Dim u As New userMainMenu()

u.UserEmail = email.Text.Trim()

u.Show()

Hide()

ElseIf role = "admin" Then

Dim d As New AdminDashboard()

d.UserEmail = email.Text.Trim()

d.Show()

Hide()

End If

End If

r.Close()

End Using

End Using

email.Clear()

password.Clear()

End Sub

Private Sub password\_KeyDown(sender As Object, e As KeyEventArgs) Handles password.KeyDown

If e.KeyCode = Keys.Enter Then

e.SuppressKeyPress = True

Button1.PerformClick()

End If

End Sub

Protected Overrides Sub OnHandleCreated(e As EventArgs)

MyBase.OnHandleCreated(e)

Me.SetStyle(ControlStyles.AllPaintingInWmPaint Or ControlStyles.UserPaint Or ControlStyles.OptimizedDoubleBuffer Or ControlStyles.ResizeRedraw, True)

Me.UpdateStyles()

End Sub

Private Sub Login\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

gif = Image.FromFile("C:\Users\UM IR PC\Documents\Parking 3rd Year\FlashDrive\Parking-Slot-Management-System-master\Parking-Slot-Management-System-master\PLFINALS\Resources\gif.gif")

Me.BackgroundImageLayout = ImageLayout.Stretch

Me.DoubleBuffered = True

ImageAnimator.Animate(gif, AddressOf OnFrameChanged)

frameTimer = New Timer() With {.Interval = 33}

AddHandler frameTimer.Tick, AddressOf UpdateBackgroundFrame

frameTimer.Start()

MakeControlsTransparent()

Button1.Focus()

FixLayout()

FixTopButtons()

End Sub

Private Sub OnFrameChanged(o As Object, e As EventArgs)

End Sub

Private Sub UpdateBackgroundFrame(sender As Object, e As EventArgs)

ImageAnimator.UpdateFrames(gif)

Invalidate(False)

End Sub

Protected Overrides Sub OnPaintBackground(pevent As PaintEventArgs)

pevent.Graphics.DrawImage(gif, Me.ClientRectangle)

End Sub

Private Sub MakeControlsTransparent()

For Each ctrl As Control In {pss, CheckBox1, Label1, Label2, Label3, Label4, PictureBox1}

ctrl.BackColor = Color.Transparent

EnableDB(ctrl)

Next

End Sub

Private Sub EnableDB(ctrl As Control)

Try

Dim m = GetType(Control).GetMethod("SetStyle", BindingFlags.Instance Or BindingFlags.NonPublic)

m.Invoke(ctrl, New Object() {ControlStyles.OptimizedDoubleBuffer Or ControlStyles.AllPaintingInWmPaint Or ControlStyles.UserPaint, True})

Catch

End Try

End Sub

Protected Overrides Sub OnFormClosing(e As FormClosingEventArgs)

If frameTimer IsNot Nothing Then frameTimer.Stop()

ImageAnimator.StopAnimate(gif, AddressOf OnFrameChanged)

MyBase.OnFormClosing(e)

End Sub

Private Sub CheckBox1\_CheckedChanged(sender As Object, e As EventArgs) Handles CheckBox1.CheckedChanged

password.UseSystemPasswordChar = Not CheckBox1.Checked

End Sub

Private Sub Label1\_Click(sender As Object, e As EventArgs) Handles Label1.Click

If Me.WindowState = FormWindowState.Normal Then

Me.WindowState = FormWindowState.Maximized

Else

Me.WindowState = FormWindowState.Normal

Me.CenterToScreen()

End If

End Sub

Private Sub Label3\_Click\_1(sender As Object, e As EventArgs) Handles Label3.Click

Me.WindowState = FormWindowState.Minimized

End Sub

Private Sub FixLayout()

Dim cx As Integer = Me.ClientSize.Width \ 2

PictureBox1.Left = cx - PictureBox1.Width \ 2

pss.Left = cx - pss.Width \ 2

email.Left = cx - email.Width \ 2

password.Left = cx - password.Width \ 2

Button1.Left = cx - Button1.Width \ 2

CheckBox1.Left = cx - (CheckBox1.Width \ -3)

Label2.Left = cx - (Label2.Width \ -10)

End Sub

Private Sub FixTopButtons()

Dim pad As Integer = 10

Dim gap As Integer = 5

Label4.Top = 5

Label1.Top = 5

Label3.Top = 5

Label4.Left = Me.ClientSize.Width - Label4.Width - pad

Label1.Left = Label4.Left - Label1.Width - gap

Label3.Left = Label1.Left - Label3.Width - gap

End Sub

Private Sub Login\_Resize(sender As Object, e As EventArgs) Handles MyBase.Resize

FixLayout()

FixTopButtons()

End Sub

End Class

**Email Confirmation and OTP Confirmation**

Imports System.Data.SqlClient

Imports System.Net

Imports System.Net.Mail

Public Class Address

Dim otpCode As String

Private countdown As Integer = 60

Private WithEvents otpTimer As New Timer()

Public Shared resetEmail As String

Public Function GenerateOTP() As String

Return New Random().Next(100000, 999999).ToString()

End Function

Public Sub SendOTP(email As String, code As String)

Try

Dim smtp As New SmtpClient("smtp.gmail.com") With {

.Port = 587,

.EnableSsl = True,

.Credentials = New NetworkCredential("joshuanombradojaculba@gmail.com", "hula miwi pgpv dvky")

}

Dim mail As New MailMessage("joshuanombradojaculba@gmail.com", email) With {

.Subject = "OTP",

.Body = code

}

smtp.Send(mail)

Catch

End Try

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

If TextBox1.Text.Trim() = "" Then Exit Sub

Dim connString As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Dim query As String = "SELECT COUNT(\*) FROM [dbo].[User] WHERE [Email]=@Email"

Using conn As New SqlConnection(connString)

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Email", TextBox1.Text.Trim())

conn.Open()

If Convert.ToInt32(cmd.ExecuteScalar()) = 0 Then

TextBox1.Clear()

Exit Sub

End If

End Using

End Using

otpCode = GenerateOTP()

SendOTP(TextBox1.Text.Trim(), otpCode)

TextBox1.Visible = False

Button1.Visible = False

Label3.Visible = False

Label2.Visible = True

TextBox2.Visible = True

Button2.Visible = True

Button3.Visible = True

Button4.Visible = False

Label4.Visible = True

countdown = 60

Button3.Text = "60"

Button3.Enabled = False

Timer1.Interval = 1000

Timer1.Start()

End Sub

Private Sub Address\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

CenterControls()

FixTopButtons()

TextBox1.Visible = True

Button1.Visible = True

Label3.Visible = True

Label2.Visible = False

TextBox2.Visible = False

Button2.Visible = False

Button3.Visible = False

Label4.Visible = False

Button4.Visible = False

End Sub

Private Sub Label3\_Click(sender As Object, e As EventArgs) Handles Label3.Click

Me.Hide()

Login.Show()

Login.email.Clear()

Login.password.Clear()

End Sub

Private Sub Label4\_Click(sender As Object, e As EventArgs) Handles Label4.Click

TextBox1.Visible = True

Button1.Visible = True

Label3.Visible = True

TextBox1.Clear()

Label2.Visible = False

TextBox2.Visible = False

Button2.Visible = False

Button3.Visible = False

Label4.Visible = False

End Sub

Private Sub Timer1\_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick

countdown -= 1

If countdown > 0 Then

Button3.Text = countdown

Else

Timer1.Stop()

Button3.Visible = False

Button4.Visible = True

End If

End Sub

Private Sub Button4\_Click(sender As Object, e As EventArgs) Handles Button4.Click

otpCode = GenerateOTP()

SendOTP(TextBox1.Text.Trim(), otpCode)

countdown = 60

Button3.Text = countdown

Button3.Enabled = False

Button3.Visible = True

Button4.Visible = False

Timer1.Start()

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

If TextBox2.Text.Trim() = "" Then Exit Sub

If TextBox2.Text.Trim() = otpCode Then

Timer1.Stop()

resetEmail = TextBox1.Text.Trim()

Dim f As New Newpass()

f.Label3.Text = resetEmail

f.Show()

Me.Hide()

TextBox2.Clear()

Else

TextBox2.Clear()

End If

End Sub

Private Sub TextBox1\_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged

Dim t = TextBox1.Text.Replace(" ", "")

Dim atCount = t.Count(Function(c) c = "@")

If atCount > 1 Then

Dim i = t.IndexOf("@")

t = t.Substring(0, i + 1) & t.Substring(i + 1).Replace("@", "")

End If

TextBox1.Text = t

TextBox1.SelectionStart = t.Length

End Sub

Private isHandlingTextChanged2 As Boolean = False

Private Sub TextBox2\_TextChanged(sender As Object, e As EventArgs) Handles TextBox2.TextChanged

If isHandlingTextChanged2 Then Return

isHandlingTextChanged2 = True

Dim tb = CType(sender, TextBox)

tb.ContextMenuStrip = New ContextMenuStrip()

If My.Computer.Keyboard.CtrlKeyDown AndAlso Clipboard.ContainsText() Then

tb.Text = ""

isHandlingTextChanged2 = False

Exit Sub

End If

Dim filtered = New String(tb.Text.Where(Function(c) Char.IsDigit(c)).ToArray())

If filtered.Length > 6 Then filtered = filtered.Substring(0, 6)

If tb.Text <> filtered Then

tb.Text = filtered

tb.SelectionStart = filtered.Length

End If

isHandlingTextChanged2 = False

End Sub

Private Sub TextBox2\_KeyPress(sender As Object, e As KeyPressEventArgs) Handles TextBox2.KeyPress

If Not Char.IsDigit(e.KeyChar) AndAlso e.KeyChar <> ChrW(Keys.Back) Then e.Handled = True

If TextBox2.Text.Length >= 6 AndAlso e.KeyChar <> ChrW(Keys.Back) Then e.Handled = True

End Sub

Private Sub TextBox1\_KeyPress(sender As Object, e As KeyPressEventArgs) Handles TextBox1.KeyPress

If Char.IsControl(e.KeyChar) Then Return

If Char.IsLetterOrDigit(e.KeyChar) OrElse e.KeyChar = "@"c OrElse e.KeyChar = "."c Then Return

e.Handled = True

End Sub

Private Sub TextBox1\_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox1.KeyDown

If e.KeyCode = Keys.Enter Then

e.SuppressKeyPress = True

Button1.PerformClick()

End If

End Sub

Private Sub TextBox2\_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox2.KeyDown

If e.KeyCode = Keys.Enter Then

e.SuppressKeyPress = True

Button2.PerformClick()

End If

End Sub

Private Sub Label7\_Click(sender As Object, e As EventArgs) Handles Label7.Click

If Me.WindowState = FormWindowState.Normal Then

Me.WindowState = FormWindowState.Maximized

Else

Me.WindowState = FormWindowState.Normal

Me.CenterToScreen()

End If

End Sub

Private Sub Label6\_Click(sender As Object, e As EventArgs) Handles Label6.Click

Me.WindowState = FormWindowState.Minimized

End Sub

Private Sub Address\_Resize(sender As Object, e As EventArgs) Handles MyBase.Resize

CenterControls()

FixTopButtons()

End Sub

Private Sub CenterControls()

Dim w As Integer = 500, h As Integer = 400

Dim x As Integer = (Me.ClientSize.Width - w) \ 2

Dim y As Integer = (Me.ClientSize.Height - h) \ 2

TextBox1.Left = x + 20

TextBox2.Left = x + 20

Button1.Left = x + 30

Button2.Left = x + 30

Button3.Left = x + 250

Button4.Left = x + 250

Label2.Left = x + 25

Label3.Left = x + 150

Label4.Left = x + 100

End Sub

Private Sub FixTopButtons()

Dim pad As Integer = 10

Dim gap As Integer = 5

Dim top As Integer = 5

Label5.Top = top

Label7.Top = top

Label6.Top = top

Label5.Left = Me.ClientSize.Width - Label5.Width - pad

Label7.Left = Label5.Left - Label7.Width - gap

Label6.Left = Label7.Left - Label6.Width - gap

End Sub

End Class

**Change Password**

Imports System.Data.SqlClient

Public Class Newpass

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Dim p1 As String = TextBox1.Text.Trim()

Dim p2 As String = TextBox2.Text.Trim()

If p1 = "" Or p2 = "" Then Exit Sub

If p1 <> p2 Then Exit Sub

If p1.Length < 8 Or p1.Length > 20 Then Exit Sub

Dim hasUpper = p1.Any(AddressOf Char.IsUpper)

Dim hasLower = p1.Any(AddressOf Char.IsLower)

Dim hasNum = p1.Any(AddressOf Char.IsDigit)

Dim hasSym = p1.IndexOfAny("!@#$%^&\*()\_+-=[]{}|;:'"",.<>/?".ToCharArray()) >= 0

If Not (hasUpper And hasLower And hasNum And hasSym) Then Exit Sub

Try

Using conn As New SqlConnection("Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True")

Using cmd As New SqlCommand("UPDATE [dbo].[User] SET Passwrod=@p WHERE Email=@e", conn)

cmd.Parameters.AddWithValue("@p", p1)

cmd.Parameters.AddWithValue("@e", Label3.Text.Trim())

conn.Open()

If cmd.ExecuteNonQuery() > 0 Then

TextBox1.Clear()

TextBox2.Clear()

Me.Hide()

Login.Show()

End If

End Using

End Using

Catch

End Try

End Sub

Private Sub TextBox1\_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged

CleanInputs()

End Sub

Private Sub TextBox2\_TextChanged(sender As Object, e As EventArgs) Handles TextBox2.TextChanged

CleanInputs()

End Sub

Private Sub CleanInputs()

TextBox1.Text = TextBox1.Text.Replace(" ", "")

TextBox2.Text = TextBox2.Text.Replace(" ", "")

TextBox1.SelectionStart = TextBox1.Text.Length

TextBox2.SelectionStart = TextBox2.Text.Length

TextBox1.UseSystemPasswordChar = Not CheckBox1.Checked

TextBox2.UseSystemPasswordChar = Not CheckBox1.Checked

End Sub

Private Sub CheckBox1\_CheckedChanged(sender As Object, e As EventArgs) Handles CheckBox1.CheckedChanged

TextBox1.UseSystemPasswordChar = Not CheckBox1.Checked

TextBox2.UseSystemPasswordChar = Not CheckBox1.Checked

End Sub

Private Sub Label5\_Click(sender As Object, e As EventArgs) Handles Label5.Click

Me.WindowState = If(Me.WindowState = FormWindowState.Normal, FormWindowState.Maximized, FormWindowState.Normal)

End Sub

Private Sub Label6\_Click(sender As Object, e As EventArgs) Handles Label6.Click

Me.WindowState = FormWindowState.Minimized

End Sub

Private Sub Newpass\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Label4.Anchor = AnchorStyles.Top Or AnchorStyles.Right

Label5.Anchor = AnchorStyles.Top Or AnchorStyles.Right

Label6.Anchor = AnchorStyles.Top Or AnchorStyles.Right

End Sub

Private Sub Newpass\_Resize(sender As Object, e As EventArgs) Handles MyBase.Resize

CenterControls()

End Sub

Private Sub CenterControls()

Dim w As Integer = 480, h As Integer = 300

Dim x As Integer = (Me.ClientSize.Width - w) \ 2

Dim y As Integer = (Me.ClientSize.Height - h) \ 2

Label1.Left = x

TextBox1.Left = x

Label2.Left = x

TextBox2.Left = x

Button1.Left = x

CheckBox1.Left = x + 320

Dim t As Integer = CheckBox1.Bottom + 10

Label7.Left = x : Label7.Top = t

Label8.Left = x : Label8.Top = t

Label9.Left = x : Label9.Top = t

Label10.Left = x : Label10.Top = t

Label11.Left = x : Label11.Top = t

End Sub

Private Sub TextBox1\_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox1.KeyDown

If e.KeyCode = Keys.Enter Then e.SuppressKeyPress = True : Button1.PerformClick()

End Sub

Private Sub TextBox2\_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox2\_KeyDown

If e.KeyCode = Keys.Enter Then e.SuppressKeyPress = True : Button1.PerformClick()

End Sub

End Class

**Admin Dashboard**

Imports System.Data.SqlClient

Imports System.Drawing

Imports System.Drawing.Printing

Imports System.Net

Imports System.Net.Mail

Imports System.Text.RegularExpressions

Imports System.Windows.Forms

Public Class AdminDashboard

' Private otpCode As String ' global OTP storage

Private WithEvents printDocument As New PrintDocument()

Private printPreview As New PrintPreviewDialog()

Private currentPage As Integer

Private currentRow As Integer

Public Property UserEmail As String

Private Sub Data()

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

' Base query: one row per ticket with nearest Time-Out

Dim query As String = "

SELECT

t.ID AS [Slot ID],

t.Plate AS [Plate ID],

t.Name AS [Name],

t.Valid AS [Presented ID],

FORMAT(TRY\_CONVERT(datetime, t.Duration), 'MMM d /h:mm tt') AS [Time-In],

(

SELECT TOP 1 FORMAT(TRY\_CONVERT(datetime, p.Duration), 'MMM d /h:mm tt')

FROM dbo.Payment p

WHERE p.ID = t.ID AND TRY\_CONVERT(datetime, p.Duration) >= TRY\_CONVERT(datetime, t.Duration)

ORDER BY TRY\_CONVERT(datetime, p.Duration) ASC

) AS [Time-Out]

FROM dbo.Ticket t

WHERE 1 = 1

"

' Build dynamic WHERE clause safely

Dim cmd As New SqlCommand()

cmd.Connection = conn

If DateTimePicker2.Checked AndAlso Not DateTimePicker3.Checked Then

query &= " AND CAST(TRY\_CONVERT(datetime, t.Duration) AS date) = @date1"

cmd.Parameters.AddWithValue("@date1", DateTimePicker2.Value.Date)

ElseIf DateTimePicker2.Checked AndAlso DateTimePicker3.Checked Then

query &= " AND CAST(TRY\_CONVERT(datetime, t.Duration) AS date) BETWEEN @startDate AND @endDate"

cmd.Parameters.AddWithValue("@startDate", DateTimePicker2.Value.Date)

cmd.Parameters.AddWithValue("@endDate", DateTimePicker3.Value.Date)

End If

query &= " ORDER BY TRY\_CONVERT(datetime, t.Duration) DESC"

cmd.CommandText = query

' Fill DataTable

Dim adapter As New SqlDataAdapter(cmd)

Dim dt As New DataTable()

adapter.Fill(dt)

DataGridView2.DataSource = dt

' Style DataGridView

With DataGridView2

.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill

.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells

.ScrollBars = ScrollBars.Both

.AllowUserToResizeColumns = False

.AllowUserToResizeRows = False

.AllowUserToOrderColumns = False

.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.DisableResizing

.RowHeadersVisible = False

.EnableHeadersVisualStyles = False

.AllowUserToAddRows = False

.AllowUserToDeleteRows = False

.ReadOnly = True

.BorderStyle = BorderStyle.None

.BackgroundColor = Color.White

.SelectionMode = DataGridViewSelectionMode.FullRowSelect

.DefaultCellStyle.SelectionBackColor = Color.LightBlue

.DefaultCellStyle.SelectionForeColor = Color.Black

.ColumnHeadersDefaultCellStyle.Font = New Font("Segoe UI", 10, FontStyle.Bold)

.DefaultCellStyle.Font = New Font("Segoe UI", 10)

.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

.ColumnHeadersDefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

End With

' Apply row colors

For Each row As DataGridViewRow In DataGridView2.Rows

If row.Cells("Slot ID").Value IsNot Nothing Then

Dim slotId As String = row.Cells("Slot ID").Value.ToString().ToUpper()

Dim colorToUse As Color = Color.White

If slotId.StartsWith("A") OrElse slotId.StartsWith("B") OrElse slotId.StartsWith("C") Then

colorToUse = Color.DarkGray

ElseIf slotId.StartsWith("M") Then

colorToUse = Color.Goldenrod

End If

row.DefaultCellStyle.BackColor = colorToUse

End If

Next

RemoveHandler DataGridView2.Sorted, AddressOf DataGridView2\_Sorted

AddHandler DataGridView2.Sorted, AddressOf DataGridView2\_Sorted

End Using

Catch ex As Exception

MessageBox.Show("Error loading Ticket table: " & ex.Message, "Database Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Private Sub ApplyRowColors()

For Each row As DataGridViewRow In DataGridView2.Rows

If row.Cells("Slot ID").Value IsNot Nothing Then

Dim slotId As String = row.Cells("Slot ID").Value.ToString().ToUpper()

Dim colorToUse As Color = Color.White ' default

' A, B, C share DarkGray; M uses GoldenRod

If slotId.StartsWith("A") OrElse slotId.StartsWith("B") OrElse slotId.StartsWith("C") Then

colorToUse = Color.DarkGray

ElseIf slotId.StartsWith("M") Then

colorToUse = Color.Goldenrod

End If

' Apply color (avoid blue, green, red)

If colorToUse <> Color.Red AndAlso colorToUse <> Color.Blue AndAlso colorToUse <> Color.Green Then

row.DefaultCellStyle.BackColor = colorToUse

End If

End If

Next

End Sub

Private Sub Panel2\_Paint(sender As Object, e As PaintEventArgs) Handles Panel2.Paint

End Sub

Private Sub Label12\_Click(sender As Object, e As EventArgs) Handles Label12.Click

Dim result As DialogResult = MessageBox.Show("Are you sure you want to sign out?", "Sign Out Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Question)

If result = DialogResult.Yes Then

Me.Hide()

Login.Show()

Login.email.Focus()

End If

End Sub

Private Sub Label13\_Click(sender As Object, e As EventArgs) Handles Label13.Click

Dim result As DialogResult = MessageBox.Show(

"Are you sure you want to exit the application?",

"Exit Confirmation",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question

)

If result = DialogResult.Yes Then

Application.Exit()

End If

End Sub

Private Sub Button56\_Click(sender As Object, e As EventArgs) Handles Button56.Click

Dim frm As New userMainMenu()

' Pass username to userMainMenu

frm.UserNameFromAdmin = Label28.Text.Trim

frm.Label1.Visible = True

frm.Show()

Me.Hide()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Text = "1234567890"

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

If Date.Now < DateTimePicker2.MinDate Then

DateTimePicker2.Value = DateTimePicker2.MinDate

ElseIf Date.Now > DateTimePicker2.MaxDate Then

DateTimePicker2.Value = DateTimePicker2.MaxDate

Else

DateTimePicker2.Value = Date.Now

End If

If Date.Now < DateTimePicker3.MinDate Then

DateTimePicker3.Value = DateTimePicker3.MinDate

ElseIf Date.Now > DateTimePicker3.MaxDate Then

DateTimePicker3.Value = DateTimePicker3.MaxDate

Else

DateTimePicker3.Value = Date.Now

End If

If Date.Now < DateTimePicker4.MinDate Then

DateTimePicker4.Value = DateTimePicker4.MinDate

ElseIf Date.Now > DateTimePicker4.MaxDate Then

DateTimePicker4.Value = DateTimePicker4.MaxDate

Else

DateTimePicker4.Value = Date.Now

End If

If Date.Now < DateTimePicker5.MinDate Then

DateTimePicker5.Value = DateTimePicker5.MinDate

ElseIf Date.Now > DateTimePicker5.MaxDate Then

DateTimePicker5.Value = DateTimePicker5.MaxDate

Else

DateTimePicker5.Value = Date.Now

End If

End Sub

' Email from Login

Private Sub AdminDashboard\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

' Make sure the label does not auto-size

Label28.AutoSize = False

' Set the label width to match the panel or smaller if you want margins

Label28.Width = sidePanel.Width - 20 ' leave 10px margin on each side

' Center the text inside the label

Label28.TextAlign = ContentAlignment.MiddleCenter

' Optional: adjust location to be centered vertically if needed

Label28.Left = (sidePanel.Width - Label28.Width) \ 2

Me.BackColor = Color.DimGray ' Form background

' Side panel settings

sidePanel.Dock = DockStyle.Left

' Main panels docked to fill

Panel4.Dock = DockStyle.Fill

Attendant.Dock = DockStyle.Fill

Sales.Dock = DockStyle.Fill

Car.Dock = DockStyle.Fill

' Add panels in proper Z-order

Me.Controls.Add(Panel4)

Me.Controls.Add(Attendant)

Me.Controls.Add(Sales)

Me.Controls.Add(Car)

' Show default panel

Panel4.BringToFront()

' Me.Controls.Add(CusPanel)

DateTimePicker5.MaxDate = DateTime.Today

DateTimePicker4.MaxDate = DateTime.Today

DateTimePicker3.MaxDate = DateTime.Today

DateTimePicker2.MaxDate = DateTime.Today

DateTimePicker1.MaxDate = DateTime.Today

LoadUsers()

Button8.Visible = True

Button3.Visible = False

Label21.Text = DateTime.Now.ToString("hh:mm:ss tt").ToUpper() ' Time with AM/PM

Label22.Text = DateTime.Now.ToString("MMMM dd, yyyy") ' Date format: October 28, 2025

' Start timer

Timer1.Interval = 1000 ' 1 second

Timer1.Start()

' DateTimePicker1.MaxDate = Date.Now

TextBox3.Text = "1234567890"

Panel4.Show()

Panel4.Visible = True

Panel4.BringToFront()

Attendant.SendToBack()

'Vehicle.SendToBack()

Sales.SendToBack()

Car.SendToBack()

RefreshData()

Data()

DataPayment()

LoadUsername()

If Not String.IsNullOrEmpty(UserEmail) Then

LoadUsernameFromEmail(UserEmail)

End If

End Sub

Private Sub LoadUsername()

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Using conn As New SqlConnection(connStr)

conn.Open()

Dim query As String = "SELECT Username FROM [User] WHERE Email = @Email"

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Email", UserEmail) ' ✅ use property value here

Dim username As Object = cmd.ExecuteScalar()

If username IsNot Nothing Then

Label28.Text = username.ToString()

Else

Label28.Text = "Unknown User"

End If

End Using

End Using

End Sub

Private Sub LoadUsernameFromEmail(email As String)

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Dim query As String = "SELECT Username FROM [User] WHERE Email = @Email"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Email", email)

Dim result As Object = cmd.ExecuteScalar()

If result IsNot Nothing Then

Label28.Text = result.ToString()

Label28.TextAlign = ContentAlignment.MiddleCenter

Else

Label28.Text = "Unknown User"

End If

End Using

End Using

Catch ex As Exception

MessageBox.Show("Error loading username: " & ex.Message)

End Try

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Panel4.Hide()

Panel4.Visible = False

Panel4.SendToBack()

Attendant.BringToFront()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Text = "1234567890"

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

If Date.Now < DateTimePicker2.MinDate Then

DateTimePicker2.Value = DateTimePicker2.MinDate

ElseIf Date.Now > DateTimePicker2.MaxDate Then

DateTimePicker2.Value = DateTimePicker2.MaxDate

Else

DateTimePicker2.Value = Date.Now

End If

If Date.Now < DateTimePicker3.MinDate Then

DateTimePicker3.Value = DateTimePicker3.MinDate

ElseIf Date.Now > DateTimePicker3.MaxDate Then

DateTimePicker3.Value = DateTimePicker3.MaxDate

Else

DateTimePicker3.Value = Date.Now

End If

If Date.Now < DateTimePicker4.MinDate Then

DateTimePicker4.Value = DateTimePicker4.MinDate

ElseIf Date.Now > DateTimePicker4.MaxDate Then

DateTimePicker4.Value = DateTimePicker4.MaxDate

Else

DateTimePicker4.Value = Date.Now

End If

If Date.Now < DateTimePicker5.MinDate Then

DateTimePicker5.Value = DateTimePicker5.MinDate

ElseIf Date.Now > DateTimePicker5.MaxDate Then

DateTimePicker5.Value = DateTimePicker5.MaxDate

Else

DateTimePicker5.Value = Date.Now

End If

TextBox3.Text = "1234567890"

TextBox3.ReadOnly = True

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

TextBox5.ReadOnly = False

Button8.Visible = True

Button3.Visible = False

TextBox3.Visible = False

Label20.Visible = False

CheckBox1.Visible = False

TextBox5.Visible = True

Label2.Visible = True

End Sub

Private Sub Button3\_Click(sender As Object, e As EventArgs)

Panel4.Hide()

Panel4.Visible = False

Panel4.SendToBack()

'Vehicle.BringToFront()

End Sub

Private Sub Button60\_Click(sender As Object, e As EventArgs) Handles Button60.Click

Panel4.Hide()

Panel4.Visible = False

Panel4.SendToBack()

Sales.BringToFront()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Text = "1234567890"

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

If Date.Now < DateTimePicker2.MinDate Then

DateTimePicker2.Value = DateTimePicker2.MinDate

ElseIf Date.Now > DateTimePicker2.MaxDate Then

DateTimePicker2.Value = DateTimePicker2.MaxDate

Else

DateTimePicker2.Value = Date.Now

End If

If Date.Now < DateTimePicker3.MinDate Then

DateTimePicker3.Value = DateTimePicker3.MinDate

ElseIf Date.Now > DateTimePicker3.MaxDate Then

DateTimePicker3.Value = DateTimePicker3.MaxDate

Else

DateTimePicker3.Value = Date.Now

End If

' DateTimePicker4.Value = Date.Now

' DateTimePicker5.Value = Date.Now

End Sub

Private Sub Button58\_Click(sender As Object, e As EventArgs) Handles Button58.Click

Panel4.Hide()

Panel4.Visible = False

Panel4.SendToBack()

Car.BringToFront()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Text = "1234567890"

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

If Date.Now < DateTimePicker4.MinDate Then

DateTimePicker4.Value = DateTimePicker4.MinDate

ElseIf Date.Now > DateTimePicker4.MaxDate Then

DateTimePicker4.Value = DateTimePicker4.MaxDate

Else

DateTimePicker4.Value = Date.Now

End If

If Date.Now < DateTimePicker5.MinDate Then

DateTimePicker5.Value = DateTimePicker5.MinDate

ElseIf Date.Now > DateTimePicker5.MaxDate Then

DateTimePicker5.Value = DateTimePicker5.MaxDate

Else

DateTimePicker5.Value = Date.Now

End If

End Sub

Private Sub DataPayment()

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

Dim query As String = "

SELECT

p.ID AS [Slot ID],

p.Plate AS [Plate ID],

FORMAT(TRY\_CONVERT(datetime, t.Duration), 'MMM d /h:mm tt') AS [Time-In],

FORMAT(TRY\_CONVERT(datetime, p.Duration), 'MMM d /h:mm tt') AS [Time-Out],

p.Crew AS [Crew],

p.Amount AS [Amount],

p.[Lost Ticket],

p.Late

FROM dbo.Payment p

LEFT JOIN dbo.Ticket t ON p.Plate = t.Plate

WHERE 1 = 1"

Dim cmd As New SqlCommand()

cmd.Connection = conn

' ====== DATE FILTER ======

If DateTimePicker5.Checked AndAlso Not DateTimePicker4.Checked Then

query &= " AND CAST(TRY\_CONVERT(datetime, p.Duration) AS date) = @date1"

cmd.Parameters.AddWithValue("@date1", DateTimePicker5.Value.Date)

ElseIf DateTimePicker5.Checked AndAlso DateTimePicker4.Checked Then

query &= " AND CAST(TRY\_CONVERT(datetime, p.Duration) AS date) BETWEEN @startDate AND @endDate"

cmd.Parameters.AddWithValue("@startDate", DateTimePicker5.Value.Date)

cmd.Parameters.AddWithValue("@endDate", DateTimePicker4.Value.Date)

End If

' ====== SHOW RECENT FIRST ======

query &= " ORDER BY TRY\_CONVERT(datetime, p.Duration) DESC"

cmd.CommandText = query

' ====== FILL TABLE ======

Dim adapter As New SqlDataAdapter(cmd)

Dim dt As New DataTable()

adapter.Fill(dt)

DataGridView1.DataSource = dt

End Using

' ====== TABLE STYLE ======

With DataGridView1

.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill

.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells

.ScrollBars = ScrollBars.Both

.AllowUserToResizeColumns = False

.AllowUserToResizeRows = False

.AllowUserToOrderColumns = False

.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.DisableResizing

.RowHeadersVisible = False

.EnableHeadersVisualStyles = False

.AllowUserToAddRows = False

.AllowUserToDeleteRows = False

.ReadOnly = True

.BorderStyle = BorderStyle.None

.BackgroundColor = Color.White

.SelectionMode = DataGridViewSelectionMode.FullRowSelect

.MultiSelect = False ' ✅ One-click only, no double select needed

.DefaultCellStyle.SelectionBackColor = Color.LightBlue

.DefaultCellStyle.SelectionForeColor = Color.Black

.ColumnHeadersDefaultCellStyle.Font = New Font("Segoe UI", 10, FontStyle.Bold)

.DefaultCellStyle.Font = New Font("Segoe UI", 10)

.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

.ColumnHeadersDefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

End With

' ====== COLOR ROWS ======

ApplyPaymentRewColors()

' ✅ Keep color even after sorting or filtering

RemoveHandler DataGridView1.Sorted, AddressOf DataGridView1\_Sorted

AddHandler DataGridView1.Sorted, AddressOf DataGridView1\_Sorted

RemoveHandler DataGridView1.DataBindingComplete, AddressOf DataGridView1\_DataBindingComplete

AddHandler DataGridView1.DataBindingComplete, AddressOf DataGridView1\_DataBindingComplete

' Hide unnecessary columns

DataGridView1.Columns("Lost Ticket").Visible = False

DataGridView1.Columns("Late").Visible = False

Catch ex As Exception

MessageBox.Show("Error loading Payment table: " & ex.Message, "Database Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

' ==============================

' Reapply color after sorting/filtering

' ==============================

Private Sub DataGridView1\_DataBindingComplete(sender As Object, e As DataGridViewBindingCompleteEventArgs)

ApplyPaymentRewColors()

End Sub

' ==============================

' Apply colors based on Lost Ticket / Late Fee

' ==============================

Private Sub ApplyPaymentRewColors()

For Each row As DataGridViewRow In DataGridView1.Rows

Try

Dim lostTicket As Boolean = False

Dim lateFee As Boolean = False

' Check if columns exist and not null

If row.Cells("Lost Ticket").Value IsNot DBNull.Value Then

lostTicket = Convert.ToBoolean(row.Cells("Lost Ticket").Value)

End If

If row.Cells("Late").Value IsNot DBNull.Value Then

lateFee = Convert.ToBoolean(row.Cells("Late").Value)

End If

' Default color

Dim colorToUse As Color = Color.White

' Apply colors based on values

If lostTicket AndAlso lateFee Then

colorToUse = Color.Red

ElseIf lostTicket Then

colorToUse = Color.LightSalmon

ElseIf lateFee Then

colorToUse = Color.Khaki

End If

' Apply color (excluding blue/green except red)

If colorToUse <> Color.Blue AndAlso colorToUse <> Color.Green Then

row.DefaultCellStyle.BackColor = colorToUse

End If

Catch ex As Exception

MessageBox.Show("Error loading Payment table: " & ex.Message, "Database Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

Next

End Sub

Private Sub DataGridView1\_Sorted(sender As Object, e As EventArgs)

End Sub

Private Sub DataGridView1\_CellContentClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView1.CellContentClick

DataPayment()

DataGridView1.Columns("Lost Ticket").Visible = False

DataGridView1.Columns("Late").Visible = False

ApplyPaymentRewColors()

End Sub

Private Sub DataGridView1\_Sorted\_1(sender As Object, e As EventArgs) Handles DataGridView1.Sorted

ApplyPaymentRewColors()

End Sub

Private Sub DateTimePicker5\_ValueChanged(sender As Object, e As EventArgs) Handles DateTimePicker5.ValueChanged

If DateTimePicker5.Value > DateTime.Today Then

MessageBox.Show("Future dates are not allowed.", "Invalid Date", MessageBoxButtons.OK, MessageBoxIcon.Warning)

DateTimePicker5.Value = DateTime.Today

Exit Sub

End If

DataPayment()

End Sub

Private Sub DateTimePicker4\_ValueChanged(sender As Object, e As EventArgs) Handles DateTimePicker4.ValueChanged

If DateTimePicker4.Value > DateTime.Today Then

MessageBox.Show("Future dates are not allowed.", "Invalid Date", MessageBoxButtons.OK, MessageBoxIcon.Warning)

DateTimePicker4.Value = DateTime.Today

Exit Sub

End If

DataPayment()

End Sub

Private Sub TextBox1\_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged

' --- Prevent space at the beginning ---

If TextBox1.Text.StartsWith(" ") Then

TextBox1.Text = TextBox1.Text.TrimStart()

TextBox1.SelectionStart = TextBox1.Text.Length

Exit Sub

End If

' --- Limit input length ---

If TextBox1.Text.Length > 20 Then

TextBox1.Text = TextBox1.Text.Substring(0, 20)

TextBox1.SelectionStart = TextBox1.Text.Length

Exit Sub

End If

' --- Refresh Payment table based on search ---

LoadPaymentSearch(TextBox1.Text.Trim())

End Sub

Private Sub LoadPaymentSearch(searchText As String)

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

Dim query As String = "

SELECT

p.ID AS [Slot ID],

p.Plate AS [Plate ID],

FORMAT(TRY\_CONVERT(datetime, t.Duration), 'MMM d / h:mm tt') AS [Time-In],

FORMAT(TRY\_CONVERT(datetime, p.Duration), 'MMM d / h:mm tt') AS [Time-Out],

p.Crew AS [Crew],

p.Amount AS [Amount],

p.[Lost Ticket],

p.Late

FROM dbo.Payment p

LEFT JOIN dbo.Ticket t ON p.Plate = t.Plate

WHERE 1 = 1"

Dim cmd As New SqlCommand()

cmd.Connection = conn

' --- DATE FILTERS (same as DataPayment) ---

If DateTimePicker5.Checked AndAlso Not DateTimePicker4.Checked Then

query &= " AND CAST(TRY\_CONVERT(datetime, p.Duration) AS date) = @date1"

cmd.Parameters.AddWithValue("@date1", DateTimePicker5.Value.Date)

ElseIf DateTimePicker5.Checked AndAlso DateTimePicker4.Checked Then

query &= " AND CAST(TRY\_CONVERT(datetime, p.Duration) AS date) BETWEEN @startDate AND @endDate"

cmd.Parameters.AddWithValue("@startDate", DateTimePicker5.Value.Date)

cmd.Parameters.AddWithValue("@endDate", DateTimePicker4.Value.Date)

End If

' --- SEARCH FILTER ---

If Not String.IsNullOrWhiteSpace(searchText) Then

query &= " AND (p.Plate LIKE @search OR p.Crew LIKE @search OR p.ID LIKE @search)"

cmd.Parameters.AddWithValue("@search", "%" & searchText & "%")

End If

query &= " ORDER BY TRY\_CONVERT(datetime, p.Duration) DESC"

cmd.CommandText = query

' --- Fill DataGridView ---

Dim adapter As New SqlDataAdapter(cmd)

Dim dt As New DataTable()

adapter.Fill(dt)

DataGridView1.DataSource = dt

End Using

' --- Apply row colors like DataPayment ---

DataGridView1.Columns("Lost Ticket").Visible = False

DataGridView1.Columns("Late").Visible = False

ApplyPaymentRewColors()

Catch ex As Exception

MessageBox.Show("Error loading Payment data: " & ex.Message, "Database Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

Try

' ✅ Check if DataGridView1 has rows

If DataGridView1.Rows.Count = 0 Then

MessageBox.Show("No payment data available to print.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Exit Sub

End If

' Ask for print confirmation

Dim result As DialogResult = MessageBox.Show("Do you want to print this payment report?", "Print Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Question)

If result = DialogResult.No Then

MessageBox.Show("Printing cancelled.", "Cancelled", MessageBoxButtons.OK, MessageBoxIcon.Information)

Exit Sub

End If

currentRow = 0

currentPage = 1

' ✅ Confirmation message before showing preview

MessageBox.Show("Preparing print preview. Please wait...", "Print Preview", MessageBoxButtons.OK, MessageBoxIcon.Information)

printPreview.Document = printDocumentPayment

printPreview.WindowState = FormWindowState.Maximized

printPreview.PrintPreviewControl.Zoom = 1.0

printPreview.ShowDialog()

Catch ex As Exception

MessageBox.Show("Error while showing print preview: " & ex.Message, "Print Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Private WithEvents printDocumentPayment As New Printing.PrintDocument

' At the top of your form

Private runningTotal As Decimal = 0D

Private Sub printDocumentPayment\_PrintPage(sender As Object, e As Printing.PrintPageEventArgs) Handles printDocumentPayment.PrintPage

Dim g As Graphics = e.Graphics

Dim leftMargin As Integer = e.MarginBounds.Left

Dim topMargin As Integer = e.MarginBounds.Top

Dim pageWidth As Integer = e.MarginBounds.Width

Dim bottomMargin As Integer = e.MarginBounds.Bottom

Dim fontCompany As New Font("Segoe UI", 18, FontStyle.Bold)

Dim fontAddress As New Font("Segoe UI", 12, FontStyle.Bold)

Dim fontTitle As New Font("Segoe UI", 13, FontStyle.Bold)

Dim fontHeader As New Font("Segoe UI", 11, FontStyle.Bold)

Dim fontCell As New Font("Segoe UI", 8.5)

Dim fontFooter As New Font("Segoe UI", 8, FontStyle.Italic)

Dim fntInfo As New Font("Segoe UI", 10, FontStyle.Regular)

Dim brush As Brush = Brushes.Black

Dim cellPadding As Integer = 4

Dim cellHeight As Integer = 28

Dim x As Integer = leftMargin

Dim y As Integer = topMargin

' ===== LOGO =====

If PictureBox1.Image IsNot Nothing Then

Dim logoWidth As Integer = 100

Dim logoHeight As Integer = 120

g.DrawImage(PictureBox1.Image, leftMargin, topMargin + 10, logoWidth, logoHeight)

End If

' ===== COMPANY NAME =====

g.DrawString("ParkZone Solutions", fontCompany, brush,

New RectangleF(leftMargin + 100, topMargin + 20, pageWidth - 100, 40),

New StringFormat() With {.Alignment = StringAlignment.Near, .LineAlignment = StringAlignment.Center})

' ===== COMPANY ADDRESS =====

g.DrawString("672, P.T. Anido Building, 1008 Delos Santos St, Sampaloc, Manila, 1008 Metro Manila", fontAddress, brush,

New RectangleF(leftMargin + 100, topMargin + 60, pageWidth - 100, 50),

New StringFormat() With {.Alignment = StringAlignment.Near, .LineAlignment = StringAlignment.Near})

' ===== EMAIL & CONTACT =====

Dim emailLine As String = "Email: parkzonesolutions@gmail.com"

Dim contactLine As String = "Contact: 09282446963"

g.DrawString(emailLine, fntInfo, brush,

New RectangleF(leftMargin + 100, topMargin + 100, pageWidth - 100, 25),

New StringFormat() With {.Alignment = StringAlignment.Near, .LineAlignment = StringAlignment.Center})

g.DrawString(contactLine, fntInfo, brush,

New RectangleF(leftMargin + 100, topMargin + 115, pageWidth - 100, 25),

New StringFormat() With {.Alignment = StringAlignment.Near, .LineAlignment = StringAlignment.Center})

y = topMargin + 140

' ===== REPORT TITLE =====

g.DrawString("Sales Report", fontTitle, brush, New RectangleF(leftMargin, y, pageWidth, 30),

New StringFormat() With {.Alignment = StringAlignment.Center})

y += 30

' ===== DATE RANGE =====

Dim dateRange As String = $"From {DateTimePicker5.Value:MMMM dd, yyyy} to {DateTimePicker4.Value:MMMM dd, yyyy}"

g.DrawString(dateRange, New Font("Segoe UI", 10, FontStyle.Italic), brush,

New RectangleF(leftMargin, y, pageWidth, 25),

New StringFormat() With {.Alignment = StringAlignment.Center})

y += 25

' ===== COLOR LEGEND =====

Dim legendY As Integer = y + 10

Dim totalLegendWidth As Integer = 400

Dim pageCenterX As Integer = e.PageBounds.Width \ 2

Dim legendStartX As Integer = leftMargin + ((pageWidth - totalLegendWidth) \ 2) + 50

' Lost Ticket

g.FillRectangle(New SolidBrush(Color.LightSalmon), legendStartX, legendY, 20, 15)

g.DrawRectangle(Pens.Black, legendStartX, legendY, 20, 15)

g.DrawString("Lost Ticket", fontCell, brush, legendStartX + 25, legendY - 1)

' Late Fee

g.FillRectangle(New SolidBrush(Color.Khaki), legendStartX + 130, legendY, 20, 15)

g.DrawRectangle(Pens.Black, legendStartX + 130, legendY, 20, 15)

g.DrawString("Late Fee", fontCell, brush, legendStartX + 155, legendY - 1)

' Both

g.FillRectangle(New SolidBrush(Color.Red), legendStartX + 240, legendY, 20, 15)

g.DrawRectangle(Pens.Black, legendStartX + 240, legendY, 20, 15)

g.DrawString("Both", fontCell, brush, legendStartX + 265, legendY - 1)

y = legendY + 30

' ===== CALCULATE COLUMN WIDTHS =====

Dim totalGridWidth As Integer = 0

Dim colWidths(DataGridView1.Columns.Count - 1) As Integer

For i As Integer = 0 To DataGridView1.Columns.Count - 1

If DataGridView1.Columns(i).Visible Then

colWidths(i) = Math.Max(90, DataGridView1.Columns(i).Width)

totalGridWidth += colWidths(i)

End If

Next

Dim scaleFactor As Double = Math.Min(1.0, pageWidth / totalGridWidth)

For i As Integer = 0 To colWidths.Length - 1

colWidths(i) = CInt(colWidths(i) \* scaleFactor)

Next

' ===== COLUMN HEADERS =====

x = leftMargin

For i As Integer = 0 To DataGridView1.Columns.Count - 1

If DataGridView1.Columns(i).Visible Then

g.FillRectangle(Brushes.LightGray, x, y, colWidths(i), cellHeight)

g.DrawRectangle(Pens.Black, x, y, colWidths(i), cellHeight)

g.DrawString(DataGridView1.Columns(i).HeaderText, fontHeader, brush,

New RectangleF(x + cellPadding, y, colWidths(i) - cellPadding \* 2, cellHeight),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

x += colWidths(i)

End If

Next

y += cellHeight

' ===== ROW DATA =====

Dim totalAmount As Decimal = 0D

While currentRow < DataGridView1.Rows.Count

Dim row As DataGridViewRow = DataGridView1.Rows(currentRow)

If row.Visible Then

x = leftMargin

Dim rowColor As Color = If(row.DefaultCellStyle.BackColor.IsEmpty, Color.White, row.DefaultCellStyle.BackColor)

For i As Integer = 0 To DataGridView1.Columns.Count - 1

If DataGridView1.Columns(i).Visible Then

Dim cellValue As String = If(row.Cells(i).Value IsNot Nothing, row.Cells(i).Value.ToString(), "")

If DataGridView1.Columns(i).HeaderText = "Amount" Then

Dim val As Decimal

If Decimal.TryParse(cellValue, val) Then

totalAmount += val ' page total

runningTotal += val ' cumulative total across pages

End If

End If

Using backBrush As New SolidBrush(rowColor)

g.FillRectangle(backBrush, x, y, colWidths(i), cellHeight)

End Using

g.DrawRectangle(Pens.Black, x, y, colWidths(i), cellHeight)

g.DrawString(cellValue, fontCell, Brushes.Black,

New RectangleF(x + cellPadding, y, colWidths(i) - cellPadding \* 2, cellHeight),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

x += colWidths(i)

End If

Next

y += cellHeight

' ===== PAGE BREAK DETECTION =====

If y + cellHeight > bottomMargin - 80 Then

Dim footerTextBreak As String = $"Page {currentPage} Prepared by: {Label28.Text} Printed on {Date.Now:MMMM dd, yyyy hh:mm tt}"

g.DrawString(footerTextBreak, fontFooter, Brushes.Black, leftMargin, bottomMargin + 20)

e.HasMorePages = True

currentPage += 1

currentRow += 1

Return

End If

End If

currentRow += 1

End While

' ===== TOTAL ROW ===== 'CUMULATIVE TOTAL FIX

Dim fontTotal As New Font("Segoe UI", 11, FontStyle.Bold)

Dim totalY As Integer = y

x = leftMargin

For i As Integer = 0 To DataGridView1.Columns.Count - 1

If DataGridView1.Columns(i).Visible Then

g.DrawRectangle(Pens.Black, x, totalY, colWidths(i), cellHeight)

If DataGridView1.Columns(i).HeaderText = "Slot ID" Then

g.DrawString("TOTAL", fontTotal, Brushes.Black,

New RectangleF(x + cellPadding, totalY, colWidths(i) - cellPadding \* 2, cellHeight),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

ElseIf DataGridView1.Columns(i).HeaderText = "Amount" Then

g.DrawString("₱" & runningTotal.ToString("N2"), fontTotal, Brushes.Black,

New RectangleF(x + cellPadding, totalY, colWidths(i) - cellPadding \* 2, cellHeight),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

End If

x += colWidths(i)

End If

Next

' ===== FOOTER =====

Dim footerText As String = $"Page {currentPage} Prepared by: {Label28.Text} Printed on {Date.Now:MMMM dd, yyyy hh:mm tt}"

g.DrawString(footerText, fontFooter, Brushes.Black, leftMargin, bottomMargin + 20)

If currentRow < DataGridView1.Rows.Count Then

e.HasMorePages = True

Else

e.HasMorePages = False

currentRow = 0

currentPage = 1

runningTotal = 0D ' reset after all pages printed

End If

End Sub

' TextBox4: Only alphabetic characters, no numbers or special characters, cannot start with space

Private Sub TextBox4\_TextChanged(sender As Object, e As EventArgs) Handles TextBox4.TextChanged

Dim txt As TextBox = CType(sender, TextBox)

Dim caretPos As Integer = txt.SelectionStart

' Allow letters and spaces

txt.Text = Regex.Replace(txt.Text, "[^a-zA-Z ]", "")

' Prevent starting with a space

If txt.Text.StartsWith(" ") Then

txt.Text = txt.Text.TrimStart()

End If

' Prevent only spaces if the textbox is empty

If String.IsNullOrWhiteSpace(txt.Text) Then

txt.Clear()

End If

txt.SelectionStart = caretPos

End Sub

' TextBox6: Allow all except special symbols @,!,%,\*,&,^,~, cannot start with space

Private Sub TextBox6\_TextChanged(sender As Object, e As EventArgs) Handles TextBox6.TextChanged

Dim txt As TextBox = CType(sender, TextBox)

Dim caretPos As Integer = txt.SelectionStart

' Remove forbidden symbols

txt.Text = Regex.Replace(txt.Text, "[@!%\*&^~]", "")

' Prevent starting with a space

If txt.Text.StartsWith(" ") Then

txt.Text = txt.Text.TrimStart()

End If

' txt.SelectionStart = caretPos

End Sub

Private Sub Button10\_Click(sender As Object, e As EventArgs) Handles Button10.Click

If String.IsNullOrWhiteSpace(TextBox5.Text) Then

MessageBox.Show("Please select a user to delete.", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

Exit Sub

End If

Try

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Using conn As New SqlConnection(connStr)

conn.Open()

Dim role As String = ""

Dim userEmail As String = ""

Using cmd As New SqlCommand("SELECT Email, Role FROM [User] WHERE Username=@Username", conn)

cmd.Parameters.AddWithValue("@Username", TextBox5.Text.Trim())

Using reader = cmd.ExecuteReader()

If reader.Read() Then

userEmail = reader("Email").ToString()

role = reader("Role").ToString()

Else

MessageBox.Show("User not found.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Exit Sub

End If

End Using

End Using

If role.ToLower() = "admin" Then

MessageBox.Show("Cannot delete Admin account.", "Action Denied", MessageBoxButtons.OK, MessageBoxIcon.Error)

Exit Sub

End If

If MessageBox.Show("Are you sure you want to delete this user?", "Confirm Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question) = DialogResult.No Then

' Reset your form controls as needed

TextBox3.Text = "1234567890"

TextBox3.ReadOnly = True

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

TextBox5.ReadOnly = False

Button8.Visible = True

Button3.Visible = False

TextBox3.Visible = False

Label20.Visible = False

CheckBox1.Visible = False

TextBox5.Visible = True

Label2.Visible = True

Exit Sub

End If

' ----------------------

' Get admin email dynamically

Dim adminEmail As String = ""

Using cmd As New SqlCommand("SELECT TOP 1 Email FROM [User] WHERE Role='Admin'", conn)

Using reader = cmd.ExecuteReader()

If reader.Read() Then

adminEmail = reader("Email").ToString()

Else

MessageBox.Show("No admin account found for OTP verification.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Exit Sub

End If

End Using

End Using

' Admin OTP before deletion

If Not VerifyAdminOTP(adminEmail) Then Exit Sub

' Perform deletion

DeleteUser(TextBox5.Text.Trim())

SendGoodbyeEmail(userEmail)

MessageBox.Show("User deleted successfully!", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information)

' Clear form

TextBox3.Text = "1234567890"

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

LoadUsers()

TextBox3.Visible = False

Label20.Visible = False

CheckBox1.Visible = False

TextBox5.Visible = True

Label2.Visible = True

' ✅ Confirmation message (added)

MessageBox.Show("Action completed successfully.", "Confirmation", MessageBoxButtons.OK, MessageBoxIcon.Information)

End Using

Catch ex As Exception

MessageBox.Show("Error deleting user: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

' For printing

Private WithEvents printDocumentUser As New Printing.PrintDocument

Private currentRowUser As Integer = 0

Private currentPageUser As Integer = 1

' ===== GLOBAL VARIABLES (Place at the top of your form) =====

' ===== BUTTON CLICK: PRINT PREVIEW =====

Private Sub Button11\_Click(sender As Object, e As EventArgs) Handles Button11.Click

Try

' Ask for print confirmation

Dim result As DialogResult = MessageBox.Show("Do you want to print this report?", "Print Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Question)

If result = DialogResult.No Then

MessageBox.Show("Printing cancelled.", "Cancelled", MessageBoxButtons.OK, MessageBoxIcon.Information)

Exit Sub

End If

' Reset pagination counters

currentRowUser = 0

currentPageUser = 1

' Confirmation message before showing preview

MessageBox.Show("Preparing print preview. Please wait...", "Print Preview", MessageBoxButtons.OK, MessageBoxIcon.Information)

printPreview.Document = printDocumentUser

printPreview.WindowState = FormWindowState.Maximized

printPreview.PrintPreviewControl.Zoom = 1.0

printPreview.ShowDialog()

Catch ex As Exception

MessageBox.Show("Error while showing print preview: " & ex.Message, "Print Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

' ===== PRINT PAGE EVENT =====

Private Sub printDocumentUser\_PrintPage(sender As Object, e As Printing.PrintPageEventArgs) Handles printDocumentUser.PrintPage

Dim g As Graphics = e.Graphics

' ===== PAGE SETTINGS =====

e.PageSettings.Landscape = False

Dim leftMargin As Integer = e.MarginBounds.Left + 60

Dim topMargin As Integer = e.MarginBounds.Top

Dim bottomMargin As Integer = e.MarginBounds.Bottom

Dim pageWidth As Integer = e.MarginBounds.Width - 120

Dim brush As Brush = Brushes.Black

Dim y As Integer = topMargin

' ===== FONTS =====

Dim fontCompany As New Font("Segoe UI", 20, FontStyle.Bold)

Dim fontAddress As New Font("Arial", 8)

Dim fontHeader As New Font("Segoe UI", 9, FontStyle.Bold)

Dim fontCell As New Font("Segoe UI", 8)

Dim fontTitle As New Font("Segoe UI", 12, FontStyle.Bold)

Dim fontFooter As New Font("Arial", 8, FontStyle.Italic)

Dim fontTotal As New Font("Segoe UI", 12, FontStyle.Bold)

' ===== LOGO =====

If PictureBox1.Image IsNot Nothing Then

g.DrawImage(PictureBox1.Image, leftMargin - 40, topMargin + 10, 90, 95)

End If

' ===== COMPANY INFO =====

Dim centerWidth As Single = e.PageBounds.Width

g.DrawString("ParkZone Solutions Inc.", fontCompany, brush,

New RectangleF(leftMargin + 60, topMargin + 10, pageWidth - 100, 40),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

Dim address As String = "672, P.T. Anido Building, M.V Delos Santos St, Sampaloc, Manila, 1008 Metro Manila"

Dim email As String = "Email: parkzonesolutions@gmail.com"

Dim contact As String = "Contact: 09282446963"

y += 60

g.DrawString(address, fontAddress, brush, (centerWidth - g.MeasureString(address, fontAddress).Width) / 2, y)

y += 15

g.DrawString(email, fontAddress, brush, (centerWidth - g.MeasureString(email, fontAddress).Width) / 2, y)

y += 15

g.DrawString(contact, fontAddress, brush, (centerWidth - g.MeasureString(contact, fontAddress).Width) / 2, y)

y += 30

' ===== REPORT TITLE =====

g.DrawString("Parking Attendant Report", fontTitle, brush,

New RectangleF(leftMargin, y, pageWidth, 25),

New StringFormat() With {.Alignment = StringAlignment.Center})

y += 35

' ===== LEGEND =====

Dim legendX As Integer = leftMargin + ((pageWidth - 135) \ 2) - 30

g.FillRectangle(Brushes.Khaki, legendX, y, 15, 15)

g.DrawRectangle(Pens.Black, legendX, y, 15, 15)

g.DrawString("Admin", fontCell, brush, legendX + 25, y)

g.FillRectangle(Brushes.LightSalmon, legendX + 80, y, 15, 15)

g.DrawRectangle(Pens.Black, legendX + 80, y, 15, 15)

g.DrawString("Parking Attendant", fontCell, brush, legendX + 105, y)

y += 30

' ===== TABLE HEADER =====

Dim colWidths() As Integer = {90, 120, 90, 80, 110, 80, 85, 90}

Dim headers() As String = {"NAME", "EMAIL", "USERNAME", "PASSWORD", "ADDRESS", "BIRTHDATE", "PHONE", "HIRED DATE"}

Dim totalTableWidth As Integer = colWidths.Sum()

Dim tableStartX As Integer = leftMargin + (pageWidth - totalTableWidth) \ 2

Dim colX As Integer = tableStartX

Dim cellHeight As Integer = 22

For i As Integer = 0 To headers.Length - 1

g.FillRectangle(Brushes.LightGray, colX, y, colWidths(i), cellHeight)

g.DrawRectangle(Pens.Black, colX, y, colWidths(i), cellHeight)

g.DrawString(headers(i), fontHeader, brush,

New RectangleF(colX, y, colWidths(i), cellHeight),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

colX += colWidths(i)

Next

y += cellHeight

' ===== FETCH DATA =====

Dim dt As New DataTable()

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Using conn As New SqlConnection(connStr)

conn.Open()

Using cmd As New SqlCommand("SELECT Name, Email, Username, Passwrod, Address, Birth, Contact, Hired, Role FROM [User] ORDER BY CASE WHEN Role='Admin' THEN 0 ELSE 1 END, Hired", conn)

Dim da As New SqlDataAdapter(cmd)

da.Fill(dt)

End Using

End Using

Dim totalCount As Integer = 0

' ===== DRAW DATA ROWS =====

While currentRowUser < dt.Rows.Count

If y + cellHeight > bottomMargin - 80 Then

Dim footerText As String = $"Page {currentPageUser} Prepared by: {Label28.Text} Printed on {Date.Now:MMMM dd, yyyy hh:mm tt}"

g.DrawString(footerText, fontFooter, brush, leftMargin, bottomMargin + 20)

e.HasMorePages = True

currentPageUser += 1

Return

End If

Dim row = dt.Rows(currentRowUser)

Dim role As String = row("Role").ToString()

Dim rowBrush As Brush = If(role = "Admin", Brushes.Khaki, Brushes.LightSalmon)

colX = tableStartX

g.FillRectangle(rowBrush, tableStartX, y, totalTableWidth, cellHeight)

For i As Integer = 0 To headers.Length - 1

Dim value As String = row(i).ToString()

If headers(i) = "BIRTHDATE" Or headers(i) = "HIRED DATE" Then

Dim d As Date

If Date.TryParse(value, d) Then

value = d.ToString("MM/dd/yyyy")

End If

End If

g.DrawRectangle(Pens.Black, colX, y, colWidths(i), cellHeight)

g.DrawString(value, fontCell, brush,

New RectangleF(colX + 2, y + 2, colWidths(i) - 4, cellHeight - 4),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

colX += colWidths(i)

Next

y += cellHeight

totalCount += 1

currentRowUser += 1

End While

' ===== TOTAL ROW =====

Dim totalY As Integer = y

g.FillRectangle(Brushes.LightBlue, tableStartX, totalY, totalTableWidth, cellHeight)

g.DrawRectangle(Pens.Black, tableStartX, totalY, totalTableWidth, cellHeight)

g.DrawString($"TOTAL RECORDS: {totalCount}", fontTotal, brush,

New RectangleF(tableStartX, totalY, totalTableWidth, cellHeight),

New StringFormat() With {.Alignment = StringAlignment.Center, .LineAlignment = StringAlignment.Center})

y += cellHeight

' ===== FOOTER (LAST PAGE) =====

Dim footerTextFinal As String = $"Page {currentPageUser} Prepared by: {Label28.Text} Printed on {Date.Now:MMMM dd, yyyy hh:mm tt}"

g.DrawString(footerTextFinal, fontFooter, brush, leftMargin, bottomMargin + 20)

e.HasMorePages = False

currentRowUser = 0

currentPageUser = 1

End Sub

Private Sub AdminDashboard\_MaximizedBoundsChanged(sender As Object, e As EventArgs) Handles MyBase.MaximizedBoundsChanged

End Sub

Private Sub AdminDashboard\_MaximumSizeChanged(sender As Object, e As EventArgs) Handles MyBase.MaximumSizeChanged

End Sub

Private Sub AdminDashboard\_Resize(sender As Object, e As EventArgs) Handles MyBase.Resize

' Show default panel

Panel4.BringToFront()

End Sub

Private Sub Button5\_Click(sender As Object, e As EventArgs)

End Sub

Private Sub AdminDashboard\_FormClosing(sender As Object, e As FormClosingEventArgs) Handles MyBase.FormClosing

If e.CloseReason = CloseReason.UserClosing Then

Dim result As DialogResult = MessageBox.Show(

"Are you sure you want to exit the application?",

"Exit Confirmation",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question

)

If result = DialogResult.No Then

e.Cancel = True ' Cancel the close event

Else

Application.Exit()

End If

End If

End Sub

' Private currentPassword As String = ""

Private Sub CheckBox1\_CheckedChanged(sender As Object, e As EventArgs) Handles CheckBox1.CheckedChanged

If CheckBox1.Checked Then

' ✅ Show the real password

TextBox3.UseSystemPasswordChar = False

' TextBox3.Text = currentPassword

Else

' ✅ Hide password again

TextBox3.UseSystemPasswordChar = True

' TextBox3.Text = currentPassword

End If

End Sub

Private Sub DataGridView4\_CellContentClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView4.CellContentClick

End Sub

Private Sub DateTimePicker1\_ValueChanged(sender As Object, e As EventArgs) Handles DateTimePicker1.ValueChanged

End Sub

Private Sub DateTimePicker1\_CloseUp(sender As Object, e As EventArgs) Handles DateTimePicker1.CloseUp

Dim selectedDate As Date = DateTimePicker1.Value

Dim today As Date = Date.Today

Dim age As Integer = today.Year - selectedDate.Year

' Adjust if the birthday hasn’t occurred yet this year

If (selectedDate > today.AddYears(-age)) Then

age -= 1

End If

' Show message only after user finishes picking (CloseUp event)

If age < 18 Then

MessageBox.Show("THAT IS AGAINST THE LAW!", "Age Restriction (Republic Act No. 7610)", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox3.Text = "1234567890"

TextBox3.ReadOnly = True

TextBox4.Clear()

TextBox5.Clear()

TextBox6.Clear()

TextBox7.Clear()

TextBox8.Clear()

If Date.Now < DateTimePicker1.MinDate Then

DateTimePicker1.Value = DateTimePicker1.MinDate

ElseIf Date.Now > DateTimePicker1.MaxDate Then

DateTimePicker1.Value = DateTimePicker1.MaxDate

Else

DateTimePicker1.Value = Date.Now

End If

TextBox5.ReadOnly = False

Button8.Visible = True

Button3.Visible = False

TextBox3.Visible = False

Label20.Visible = False

CheckBox1.Visible = False

TextBox5.Visible = True

Label2.Visible = True

End Sub

Private Sub DateTimePicker1\_KeyDown(sender As Object, e As KeyEventArgs) Handles DateTimePicker1.KeyDown

e.SuppressKeyPress = True

End Sub

End Class

**User Dashboard**

Imports System.Data.SqlClient

Imports System.Drawing

Imports System.Drawing.Imaging

Imports System.Drawing.Printing

Imports System.Runtime.InteropServices

Imports System.Text

Imports ZXing

Imports ZXing.Common

Imports System.Windows.Forms

Public Class userMainMenu

Private WithEvents PrintDoc As New PrintDocument()

Private TicketBitmap As Bitmap

Public Property UserNameFromAdmin As String

Public Sub selectviewparking()

parkingPanel.Show()

End Sub

Private Sub OpenCameraAndScan()

Dim scannedCode As String = String.Empty

' --- Show parking panel ---

Me.Show()

parkingPanel.Visible = True

parkingPanel.BringToFront()

' --- Open camera ---

Using camForm As New CameraForm()

If camForm.ShowDialog(Me) = DialogResult.OK Then

scannedCode = camForm.ScannedCode

End If

End Using

If String.IsNullOrEmpty(scannedCode) Then Return

Try

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Using conn As New SqlConnection(connStr)

conn.Open()

' --- Get scanned ticket ---

Dim query As String = "SELECT ID, Name, Plate, Time, Duration, Status, Barcode FROM dbo.Ticket WHERE Barcode=@Barcode"

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Barcode", scannedCode)

Using reader As SqlDataReader = cmd.ExecuteReader()

If reader.Read() Then

' --- Extract ticket info ---

Dim ticketSlotID As String = reader("ID").ToString()

Dim ticketName As String = reader("Name").ToString()

Dim ticketPlate As String = reader("Plate").ToString()

Dim ticketBarcode As String = reader("Barcode").ToString()

Dim status As String = reader("Status").ToString().Trim().ToLower()

Dim allowedTime As Integer = CInt(reader("Time"))

Dim timeIn As DateTime = Convert.ToDateTime(reader("Duration"))

' --- Status check ---

If Not status.Equals("occupied", StringComparison.OrdinalIgnoreCase) Then

MessageBox.Show("This ticket cannot be used because its status is '" & reader("Status").ToString() & "'. Only 'Occupied' tickets are valid.", "Invalid Ticket", MessageBoxButtons.OK, MessageBoxIcon.Error)

parkingPanel.BringToFront()

Return

End If

' --- Populate labels BEFORE duplicate check ---

displayPanel.Visible = True

displayPanel.BringToFront()

parkingPanel.Visible = False

Label25.Text = scannedCode

Label27.Text = ticketSlotID

Label26.Text = ticketPlate

Label5.Text = ticketName

Label8.Text = allowedTime.ToString()

Label40.Visible = False

Label39.Visible = False

' --- Duration calculation ---

Dim nowTime As DateTime = DateTime.Now

Dim totalDuration As TimeSpan = nowTime - timeIn

Dim totalDays As Double = totalDuration.TotalDays

Dim years As Integer = CInt(Math.Floor(totalDays / 365))

Dim months As Integer = CInt(Math.Floor((totalDays Mod 365) / 30))

Dim days As Integer = CInt(Math.Floor(totalDays Mod 30))

Dim hours As Integer = CInt(Math.Floor(totalDuration.TotalHours Mod 24))

Dim minutes As Integer = CInt(Math.Floor(totalDuration.TotalMinutes Mod 60))

Dim displayDuration As String = ""

If years > 0 Then displayDuration &= years & "Yr "

If months > 0 Then displayDuration &= months & "Mo "

If days > 0 Then displayDuration &= days & "D "

If hours > 0 Then displayDuration &= hours & "Hr "

If minutes > 0 Then displayDuration &= minutes & "Min"

If displayDuration = "" Then displayDuration = "0Min"

Label24.Text = displayDuration.Trim()

' --- Late fee ---

Dim allowedMinutes As Double = allowedTime \* 60

Dim extraMinutes As Double = totalDuration.TotalMinutes - allowedMinutes

Dim lateFee As Decimal = If(extraMinutes > 0, extraMinutes \* 2D, 0D)

Label6.Text = "PHP " & lateFee.ToString("0.00")

' --- Total ---

Dim baseAmount As Decimal = allowedTime \* 50

Label22.Text = "PHP " & (baseAmount + lateFee).ToString("0.00")

reader.Close() ' Close reader after all fields are used

' --- Duplicate check (ignore current barcode) ---

Dim duplicateCheckQuery As String =

"SELECT COUNT(\*) FROM dbo.Ticket

WHERE Status='Available'

AND (ID=@ID OR Name=@Name OR Plate=@Plate OR Barcode=@Barcode)

AND Barcode <> @Barcode"

Using dupCmd As New SqlCommand(duplicateCheckQuery, conn)

dupCmd.Parameters.AddWithValue("@ID", ticketSlotID)

dupCmd.Parameters.AddWithValue("@Name", ticketName)

dupCmd.Parameters.AddWithValue("@Plate", ticketPlate)

dupCmd.Parameters.AddWithValue("@Barcode", ticketBarcode)

Dim duplicateCount As Integer = CInt(dupCmd.ExecuteScalar())

If duplicateCount > 0 Then

' MessageBox.Show("A similar active ticket exists with 'Available' status. Please verify the slot or ticket details.", "Validation Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning)

parkingPanel.BringToFront()

Return

End If

End Using

' --- Validate slot ---

Dim slotId As String = Label2.Text

If ticketSlotID <> slotId Then

MessageBox.Show("This QR code does not belong to the selected slot.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

parkingPanel.BringToFront()

Return

End If

Else

MessageBox.Show("Ticket not detected. Camera refresh in progress to maintain system efficiency.", "System Optimization", MessageBoxButtons.OK, MessageBoxIcon.Warning)

parkingPanel.BringToFront()

Return

End If

End Using

End Using

End Using

Catch ex As Exception

MessageBox.Show("Error loading ticket: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Public Sub selectavailticket()

availPanel.Show()

End Sub

Public Sub selectlatestreceipt()

displayPanel.Show()

End Sub

Public Property UserEmail As String

Private Sub userMainMenu\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

' Ensure the label is inside the correct panel

Label28.AutoSize = False

' Set the label width smaller than panel to leave margins

Label28.Width = sidePanel.ClientSize.Width - 20 ' client size for accurate width

' Center text inside label

Label28.TextAlign = ContentAlignment.MiddleCenter

' Center the label itself inside the panel

Label28.Left = (sidePanel.ClientSize.Width - Label28.Width) \ 2

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox4.Clear()

Me.BackColor = Color.DimGray ' Form background

' Side panel settings

sidePanel.Dock = DockStyle.Left

' CusPanel fills the rest

CusPanel.Dock = DockStyle.Fill

availPanel.Dock = DockStyle.Fill

parkingPanel.Dock = DockStyle.Fill

displayPanel.Dock = DockStyle.Fill

Me.Controls.Add(CusPanel)

ComboBox1.DropDownStyle = ComboBoxStyle.DropDownList

LoadParkingRecords() ' Load the table

InitializeVerifyPanel()

If TextBox2.Text = "" Then

Label4.Text = "0"

End If

parkingPanel.Visible = True

parkingPanel.BringToFront()

selectviewparking()

Me.DoubleBuffered = True

' Load logo into PictureBox1 from a file

' PictureBox1.Image = Image.FromFile("C:\Users\ADMIN\OneDrive\Documents\SYSTEMS (3RD YEAR)\SYSTEM-1ST SEM\HCI 101\Parking-Slot-Management-System-master\PLFINALS\Resources\quinta.png")

'PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

Using cmd As New SqlCommand("SELECT ID, Status FROM dbo.Ticket", conn)

Using reader As SqlDataReader = cmd.ExecuteReader()

While reader.Read()

Dim slotID As String = reader("ID").ToString()

Dim status As String = reader("Status").ToString()

' Update button color

For Each ctrl As Control In parkingPanel.Controls

If TypeOf ctrl Is Button Then

Dim btn As Button = CType(ctrl, Button)

If btn.Text = slotID OrElse btn.Name = "btn" & slotID Then

If status = "Occupied" Then

btn.BackColor = Color.Red

btn.ForeColor = Color.White

Else

btn.BackColor = Color.LawnGreen

btn.ForeColor = Color.Black

End If

Exit For

End If

End If

Next

End While

End Using

End Using

End Using

Catch ex As Exception

MessageBox.Show("Error loading slot status: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

Try

Dim query As String = "

WITH LatestTickets AS (

SELECT

t.ID,

t.Name,

t.Plate,

t.Duration,

t.Barcode,

ROW\_NUMBER() OVER (PARTITION BY t.ID ORDER BY t.Duration DESC) AS rn

FROM dbo.Ticket t

WHERE t.Status = 'Occupied'

)

SELECT

ID AS [SLOT ID],

Name AS [NAME],

Plate AS [PLATE NUMBER],

FORMAT(Duration, 'MM/dd/yyyy hh:mm tt') AS [TIME-IN],

Barcode

FROM LatestTickets

WHERE rn = 1

ORDER BY ID

"

Using conn As New SqlConnection(connStr)

Using da As New SqlDataAdapter(query, conn)

Dim dt As New DataTable()

da.Fill(dt)

dgvParkingRecords.DataSource = dt

End Using

End Using

' Table appearance settings

With dgvParkingRecords

.EnableHeadersVisualStyles = False

.ColumnHeadersDefaultCellStyle.Font = New Font("Times New Roman", 11, FontStyle.Bold)

.ColumnHeadersDefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

.RowHeadersVisible = False

.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill

.AllowUserToResizeRows = False

.ScrollBars = ScrollBars.Both

.GridColor = Color.Black

End With

Catch ex As Exception

MessageBox.Show("Error loading parking data: " & ex.Message, "Database Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

If Not String.IsNullOrEmpty(UserEmail) Then

LoadUserInfo(UserEmail)

End If

If Not String.IsNullOrEmpty(UserNameFromAdmin) Then

Label28.Text = UserNameFromAdmin

Label28.TextAlign = ContentAlignment.MiddleCenter

' Load corresponding email

LoadEmailByUsername(UserNameFromAdmin)

End If

Label29.Text = DateTime.Now.ToString("hh:mm:ss tt").ToUpper() ' Time with AM/PM

' Start timer

Timer1.Interval = 1000 ' 1 second

Timer1.Start()

End Sub

Public Sub LoadUserInfo(userEmail As String)

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Dim query As String = "SELECT Username FROM [User] WHERE Email = @Email"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Email", userEmail)

Dim username As Object = cmd.ExecuteScalar()

If username IsNot Nothing Then

Label28.Text = username.ToString() ' ✅ Username

Label28.TextAlign = ContentAlignment.MiddleCenter

' After loading username, get email using that username

LoadEmailByUsername(Label28.Text)

Else

Label28.Text = "Unknown User"

Label36.Text = "No email found"

End If

End Using

End Using

Catch ex As Exception

MessageBox.Show("Error loading user info: " & ex.Message)

End Try

End Sub

' 🆕 New sub to get email using the username from Label28

Private Sub LoadEmailByUsername(username As String)

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Dim query As String = "SELECT Email FROM [User] WHERE Username = @Username"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@Username", username)

Dim result As Object = cmd.ExecuteScalar()

If result IsNot Nothing Then

Label36.Text = result.ToString()

Label36.TextAlign = ContentAlignment.MiddleCenter

Else

Label36.Text = "No Email Found"

End If

End Using

End Using

Catch ex As Exception

MessageBox.Show("Error loading email: " & ex.Message)

End Try

End Sub

Private Sub Label13\_Click(sender As Object, e As EventArgs) Handles Label13.Click

Dim result As DialogResult = MessageBox.Show(

"Are you sure you want to exit the application?",

"Exit Confirmation",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question

)

If result = DialogResult.Yes Then

Application.Exit()

End If

End Sub

Private Sub viewParking\_Click(sender As Object, e As EventArgs)

selectviewparking()

End Sub

Private Sub LoadParkingRecords()

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

' ✅ Include Barcode but show only latest occupied record per slot

Dim query As String = "

WITH LatestTickets AS (

SELECT

t.ID,

t.Name,

t.Plate,

t.Duration,

t.Barcode,

ROW\_NUMBER() OVER (PARTITION BY t.ID ORDER BY t.Duration DESC) AS rn

FROM dbo.Ticket t

WHERE t.Status = 'Occupied'

)

SELECT

ID AS [SLOT ID],

Name AS [NAME],

Plate AS [PLATE NUMBER],

FORMAT(Duration, 'MM/dd/yyyy hh:mm tt') AS [TIME-IN],

Barcode

FROM LatestTickets

WHERE rn = 1

ORDER BY ID

"

Using da As New SqlDataAdapter(query, conn)

Dim dt As New DataTable()

da.Fill(dt)

dgvParkingRecords.DataSource = dt

End Using

End Using

' === DataGridView Appearance ===

With dgvParkingRecords

.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill

.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.DisplayedCells

.AllowUserToResizeColumns = False

.AllowUserToResizeRows = False

.RowHeadersVisible = False

.BackgroundColor = Color.White

.BorderStyle = BorderStyle.None

.CellBorderStyle = DataGridViewCellBorderStyle.SingleHorizontal

.GridColor = Color.Gray

.EnableHeadersVisualStyles = False

.ColumnHeadersDefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

.ColumnHeadersDefaultCellStyle.Font = New Font("Times New Roman", 11, FontStyle.Bold)

.ColumnHeadersDefaultCellStyle.BackColor = Color.Gainsboro

.ColumnHeadersDefaultCellStyle.ForeColor = Color.Black

.ColumnHeadersHeight = 35

.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter

.DefaultCellStyle.Font = New Font("Segoe UI", 10, FontStyle.Regular)

.DefaultCellStyle.BackColor = Color.White

.DefaultCellStyle.ForeColor = Color.Black

.DefaultCellStyle.SelectionBackColor = Color.LightSkyBlue

.DefaultCellStyle.SelectionForeColor = Color.Black

.ReadOnly = True

.SelectionMode = DataGridViewSelectionMode.FullRowSelect

.AllowUserToAddRows = False

' Hide the Barcode column from the grid but keep it accessible

If .Columns.Contains("Barcode") Then

.Columns("Barcode").Visible = False

End If

End With

' === Row color styling ===

For Each row As DataGridViewRow In dgvParkingRecords.Rows

If Not row.IsNewRow Then

Dim slotID As String = row.Cells("SLOT ID").Value.ToString().ToUpper()

' Gray rows

If (slotID Like "A10[1-9]" OrElse slotID Like "A11[0-2]") OrElse

(slotID Like "B20[1-9]" OrElse slotID Like "B21[0-2]") OrElse

(slotID Like "C30[1-9]" OrElse slotID Like "C31[0-2]") Then

row.DefaultCellStyle.BackColor = Color.DarkGray

row.DefaultCellStyle.ForeColor = Color.Black

' Yellow rows

ElseIf slotID Like "M00[1-9]" OrElse slotID Like "M01[0-9]" Then

row.DefaultCellStyle.BackColor = Color.Goldenrod

row.DefaultCellStyle.ForeColor = Color.Black

End If

End If

Next

Catch ex As Exception

MessageBox.Show("Error loading occupied slots: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Private Sub PerformLostTicketProcess()

Try

' 1) Ensure a row is selected

If dgvParkingRecords.CurrentRow Is Nothing Then

MessageBox.Show("Please select a record first.", "No Selection", MessageBoxButtons.OK, MessageBoxIcon.Warning)

Return

End If

' 2) Gather data

Dim slotId As String = dgvParkingRecords.CurrentRow.Cells("SLOT ID").Value.ToString()

Dim plate As String = dgvParkingRecords.CurrentRow.Cells("PLATE NUMBER").Value.ToString()

Dim name As String = dgvParkingRecords.CurrentRow.Cells("NAME").Value.ToString()

Dim timeIn As String = dgvParkingRecords.CurrentRow.Cells("TIME-IN").Value.ToString()

Dim barcodeFromTextbox As String = TextBox6.Text.Trim()

If String.IsNullOrEmpty(slotId) OrElse String.IsNullOrEmpty(plate) OrElse

String.IsNullOrEmpty(name) OrElse String.IsNullOrEmpty(barcodeFromTextbox) Then

MessageBox.Show("Security check in progress to maintain system performance. Please retry verification!", "System Security", MessageBoxButtons.OK, MessageBoxIcon.Information)

txtVerifyDigit.Clear()

cboVerifyID.SelectedIndex = -1

Return

End If

' ✅ Confirmation

Dim confirm As DialogResult = MessageBox.Show(

"Are you sure you want to mark this ticket as LOST and apply the PHP 100.00 fee?",

"Confirm Lost Ticket",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question

)

If confirm = DialogResult.No Then Return

End Sub

Private Sub TextBox7\_TextChanged(sender As Object, e As EventArgs) Handles TextBox7.TextChanged

' Remove any character that is not a digit

If System.Text.RegularExpressions.Regex.IsMatch(TextBox7.Text, "[^\d]") Then

Dim selStart As Integer = TextBox7.SelectionStart

TextBox7.Text = System.Text.RegularExpressions.Regex.Replace(TextBox7.Text, "[^\d]", "")

TextBox7.SelectionStart = Math.Max(0, selStart - 1)

End If

End Sub

Private Sub Button62\_Click\_1(sender As Object, e As EventArgs) Handles Button62.Click

' Declare hours ONCE here at the top

Dim hours As Integer

' -----------------------------

' VALIDATION FIRST

' -----------------------------

If String.IsNullOrWhiteSpace(TextBox1.Text) Then

MessageBox.Show("Customer name is required.", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox1.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If String.IsNullOrWhiteSpace(TextBox3.Text) Then

MessageBox.Show("Plate number is required.", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox3.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If Not Integer.TryParse(TextBox2.Text, hours) Then

MessageBox.Show("Please enter a valid number of hours", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox2.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If hours < 3 OrElse hours > 24 Then

MessageBox.Show("Selection is limited to 3–24 hours", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox2.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If String.IsNullOrWhiteSpace(ComboBox1.Text) Then

MessageBox.Show("Please select a Valid ID before proceeding", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

ComboBox1.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If String.IsNullOrWhiteSpace(TextBox7.Text) Then

MessageBox.Show("Please enter the last 4 digits of the ID.", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox7.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If Not IsNumeric(TextBox7.Text) Then

MessageBox.Show("Last 4 digits of ID must be numeric.", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox7.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

If TextBox7.Text.Length <> 4 Then

MessageBox.Show("Please enter exactly 4 digits for the ID.", "Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Warning)

TextBox7.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

Dim result As DialogResult = MessageBox.Show("Are you sure you want to generate ticket?",

"Confirm Ticket Generation",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question)

If result = DialogResult.No Then

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

' -----------------------------

' Collect values

' -----------------------------

Dim slotId As String = Label2.Text

Dim customerName As String = TextBox1.Text

Dim plateNumber As String = TextBox3.Text

Dim validID As String = ComboBox1.Text

Dim digit As String = TextBox7.Text

Integer.TryParse(TextBox2.Text, hours)

Dim barcode As String = Label3.Text

Dim amount As Decimal

Decimal.TryParse(Label4.Text, amount)

Dim duration As DateTime = DateTime.Now

' -----------------------------

' Generate Ticket Code & QR

' -----------------------------

Try

Dim chars As String = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"

Static rand As New Random()

Dim sb As New StringBuilder()

For i As Integer = 1 To 8

sb.Append(chars(rand.Next(chars.Length)))

Next

barcode = sb.ToString()

Label3.Text = barcode

Dim qrWriter As New BarcodeWriterPixelData() With {

.Format = BarcodeFormat.QR\_CODE,

.Options = New EncodingOptions With {.Width = 200, .Height = 200, .Margin = 1}

}

Dim pixelData = qrWriter.Write(barcode)

Dim bmp As New Bitmap(pixelData.Width, pixelData.Height, PixelFormat.Format32bppArgb)

Dim bmpData As BitmapData = bmp.LockBits(New Rectangle(0, 0, bmp.Width, bmp.Height),

ImageLockMode.WriteOnly, bmp.PixelFormat)

Marshal.Copy(pixelData.Pixels, 0, bmpData.Scan0, pixelData.Pixels.Length)

bmp.UnlockBits(bmpData)

PictureBox3.SizeMode = PictureBoxSizeMode.StretchImage

If PictureBox3.Image IsNot Nothing Then

Dim oldImg = PictureBox3.Image

PictureBox3.Image = Nothing

oldImg.Dispose()

End If

PictureBox3.Image = bmp

Catch ex As Exception

MessageBox.Show("QR generation error: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Exit Sub

End Try

' -----------------------------

' Suspicious entry check

' -----------------------------

Dim connStr As String = "Data Source=DESKTOP-49ID4OC\SQLEXPRESS;Initial Catalog=Parking Management System;Integrated Security=True;TrustServerCertificate=True"

Try

Using conn As New SqlConnection(connStr)

conn.Open()

' Name occupied in another slot

Using cmd As New SqlCommand("SELECT COUNT(\*) FROM dbo.Ticket WHERE Name=@Name AND Status='Occupied' AND ID<>@SlotID", conn)

cmd.Parameters.AddWithValue("@Name", customerName)

cmd.Parameters.AddWithValue("@SlotID", slotId)

If Convert.ToInt32(cmd.ExecuteScalar()) > 0 Then

MessageBox.Show("The name '" & customerName & "' is already occupying another slot. This looks suspicious!", "Suspicious Entry", MessageBoxButtons.OK, MessageBoxIcon.Error)

TextBox1.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

End Using

' Plate occupied in another slot

Using cmd As New SqlCommand("SELECT COUNT(\*) FROM dbo.Ticket WHERE Plate=@Plate AND Status='Occupied' AND ID<>@SlotID", conn)

cmd.Parameters.AddWithValue("@Plate", plateNumber)

cmd.Parameters.AddWithValue("@SlotID", slotId)

If Convert.ToInt32(cmd.ExecuteScalar()) > 0 Then

MessageBox.Show("The plate number '" & plateNumber & "' is already occupying another slot. This looks suspicious!", "Suspicious Entry", MessageBoxButtons.OK, MessageBoxIcon.Error)

TextBox1.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

End Using

' Name + Plate combination occupied in another slot

Using cmd As New SqlCommand("SELECT COUNT(\*) FROM dbo.Ticket WHERE Name=@Name AND Plate=@Plate AND Status='Occupied' AND ID<>@SlotID", conn)

cmd.Parameters.AddWithValue("@Name", customerName)

cmd.Parameters.AddWithValue("@Plate", plateNumber)

cmd.Parameters.AddWithValue("@SlotID", slotId)

If Convert.ToInt32(cmd.ExecuteScalar()) > 0 Then

MessageBox.Show("This name and plate number combination is already occupying another slot. This looks suspicious!", "Suspicious Entry", MessageBoxButtons.OK, MessageBoxIcon.Error)

TextBox1.Focus()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

Exit Sub

End If

End Using

End Using

Catch ex As Exception

MessageBox.Show("Error checking for suspicious entries: " & ex.Message, "Database Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Exit Sub

End Try

' -----------------------------

' Insert ticket into database

Dim query As String = "

INSERT INTO dbo.Ticket

(ID, Name, Plate, Time, Barcode, Amount, Duration, Valid, Digit, Status)

VALUES (@ID, @Name, @Plate, @Time, @Barcode, @Amount, @Duration, @Valid, @Digit, @Status)

"

Try

Using conn As New SqlConnection(connStr)

Using cmd As New SqlCommand(query, conn)

cmd.Parameters.AddWithValue("@ID", slotId)

cmd.Parameters.AddWithValue("@Name", customerName)

cmd.Parameters.AddWithValue("@Plate", plateNumber)

cmd.Parameters.AddWithValue("@Time", hours)

cmd.Parameters.AddWithValue("@Barcode", barcode)

cmd.Parameters.AddWithValue("@Amount", amount)

cmd.Parameters.AddWithValue("@Duration", duration)

cmd.Parameters.AddWithValue("@Valid", validID)

cmd.Parameters.AddWithValue("@Digit", digit)

cmd.Parameters.AddWithValue("@Status", "Occupied")

conn.Open()

cmd.ExecuteNonQuery()

End Using

End Using

MessageBox.Show("Ticket successfully",

"Success", MessageBoxButtons.OK, MessageBoxIcon.Information)

Catch ex As Exception

MessageBox.Show("Error saving ticket: " & ex.Message,

"Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

' -----------------------------

' Update slot button color

' -----------------------------

Try

For Each ctrl As Control In parkingPanel.Controls

If TypeOf ctrl Is Button Then

Dim btn As Button = CType(ctrl, Button)

If btn.Text = slotId OrElse btn.Name = "btn" & slotId Then

btn.BackColor = Color.Red

btn.ForeColor = Color.White

btn.Enabled = True

Exit For

End If

End If

Next

Catch ex As Exception

MessageBox.Show("Error updating slot color: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

' -----------------------------

' Print Ticket

' -----------------------------

Try

TicketBitmap = GenerateTicketImage()

Dim customSize As New PaperSize("TicketTall", 290, 560)

PrintDoc.DefaultPageSettings.PaperSize = customSize

Dim preview As New PrintPreviewDialog()

preview.Document = PrintDoc

preview.PrintPreviewControl.Zoom = 1.0

preview.ShowDialog()

displayPanel.Visible = True

displayPanel.BringToFront()

parkingPanel.BringToFront()

TextBox1.Clear()

TextBox2.Clear()

TextBox3.Clear()

TextBox7.Clear()

ComboBox1.SelectedIndex = -1

For Each ctrl As Control In parkingPanel.Controls

If TypeOf ctrl Is Button Then

Dim btn As Button = CType(ctrl, Button)

If btn.Text = Label2.Text OrElse btn.Name = "btn" & Label2.Text Then

btn.BackColor = Color.Red

btn.ForeColor = Color.White

Exit For

End If

End If

Next

Catch ex As Exception

MessageBox.Show("Error showing print preview: " & ex.Message)

End Try

End Sub

' --- Generate the ticket image ---

Private Function GenerateTicketImage() As Bitmap

Dim bmpHeight As Integer = 500 ' total height of ticket

Dim headerSpace As Integer = 35 ' space from top

Dim footerSpace As Integer = 50 ' space from bottom

Dim bmp As New Bitmap(290, 550) ' Taller for all content

Using g As Graphics = Graphics.FromImage(bmp)

g.Clear(Color.White)

' --- Fonts ---

Dim fntTitle As New Font("Arial", 11, FontStyle.Bold)

Dim fntSlot As New Font("Arial", 10, FontStyle.Bold)

Dim fntName As New Font("Arial", 9, FontStyle.Bold)

Dim fntText As New Font("Arial", 9, FontStyle.Regular)

Dim fntNote As New Font("Arial", 8, FontStyle.Regular)

Dim fntBold As New Font("Arial", 8, FontStyle.Bold)

Dim fntAddress As New Font("Arial", 8, FontStyle.Regular)

Dim fntSemiBold As New Font("Arial", 8, FontStyle.Bold)

Dim y As Integer = headerSpace

' --- Logo ---

If AdminDashboard.PictureBox1.Image IsNot Nothing Then

Dim logoMaxWidth As Integer = 80

Dim logoMaxHeight As Integer = 60

Dim ratio As Double = Math.Min(logoMaxWidth / AdminDashboard.PictureBox1.Image.Width, logoMaxHeight / AdminDashboard.PictureBox1.Image.Height)

Dim logoW As Integer = CInt(AdminDashboard.PictureBox1.Image.Width \* ratio)

Dim logoH As Integer = CInt(AdminDashboard.PictureBox1.Image.Height \* ratio)

Dim xLogo As Integer = (bmp.Width - logoW) \ 2

g.DrawImage(AdminDashboard.PictureBox1.Image, xLogo, y, logoW, logoH)

y += logoH + 5

Else

g.DrawString("(No Logo)", fntText, Brushes.Gray, 10, y)

y += 20

End If

' --- Title ---

Dim titleText As String = "ParkZone Ticket"

Dim titleX As Single = (bmp.Width - g.MeasureString(titleText, fntTitle).Width) / 2

g.DrawString(titleText, fntTitle, Brushes.Black, titleX, y)

y += CInt(g.MeasureString(titleText, fntTitle).Height) + 10

' --- Address (Centered) ---

Dim addressLine1 As String = "672, P.T. Anido Building, M.V Delos Santos St,"

Dim addressLine2 As String = "Sampaloc, Manila, 1008 Metro Manila"

Dim emailLine As String = "Email: parkzonesolutions@gmail.com"

Dim contactLine As String = "Contact: 09282446963"

Dim addrX1 As Single = (bmp.Width - g.MeasureString(addressLine1, fntAddress).Width) / 2

g.DrawString(addressLine1, fntAddress, Brushes.Black, addrX1, y)

y += CInt(g.MeasureString(addressLine1, fntAddress).Height) + 2

Dim addrX2 As Single = (bmp.Width - g.MeasureString(addressLine2, fntAddress).Width) / 2

g.DrawString(addressLine2, fntAddress, Brushes.Black, addrX2, y)

y += CInt(g.MeasureString(addressLine2, fntAddress).Height) + 2

' --- Email and Contact (Centered, Semi-Bold) ---

Dim emailX As Single = (bmp.Width - g.MeasureString(emailLine, fntSemiBold).Width) / 2

g.DrawString(emailLine, fntSemiBold, Brushes.Black, emailX, y)

y += CInt(g.MeasureString(emailLine, fntSemiBold).Height) + 2

Dim contactX As Single = (bmp.Width - g.MeasureString(contactLine, fntSemiBold).Width) / 2

g.DrawString(contactLine, fntSemiBold, Brushes.Black, contactX, y)

y += CInt(g.MeasureString(contactLine, fntSemiBold).Height) + 10

' --- Slot ID ---

Dim slotText As String = "Slot: " & Label2.Text

Dim slotX As Single = (bmp.Width - g.MeasureString(slotText, fntSlot).Width) / 2

g.DrawString(slotText, fntSlot, Brushes.Black, slotX, y)

y += CInt(g.MeasureString(slotText, fntSlot).Height) + 5

' --- Customer Name ---

g.DrawString("Name: " & TextBox1.Text, fntName, Brushes.Black, 10, y)

y += 20

' --- Hours & Amount ---

g.DrawString("Hours: " & TextBox2.Text, fntText, Brushes.Black, 10, y)

y += 20

g.DrawString("Amount: Php " & Label4.Text, fntText, Brushes.Black, 10, y)

y += 20

' ✅ Presented ID line

g.DrawString("Presented ID: " & ComboBox1.Text, fntText, Brushes.Black, 10, y)

y += 20

' --- Date ---

g.DrawString("Entry Time: " & DateTime.Now.ToString("MMM. d, yyyy — h:mm tt"), fntText, Brushes.Black, 10, y)

y += 30

' --- QR Code (Centered) ---

Dim qrBottom As Integer = y

If PictureBox3.Image IsNot Nothing Then

Dim qrW As Integer = 100

Dim qrH As Integer = 100

Dim xQr As Integer = (bmp.Width - qrW) \ 2

g.DrawImage(PictureBox3.Image, New Rectangle(xQr, y, qrW, qrH))

qrBottom = y + qrH

End If

' --- Reminder Text (Centered) ---

Dim hrs As Integer

Integer.TryParse(TextBox2.Text, hrs)

Dim dueTime As DateTime = DateTime.Now.AddHours(hrs)

Dim reminderLines As String() = {

"Reminder: Your parking ticket must be settled",

"BEFORE " + dueTime.ToString("hh:mm tt"),

"Please avoid penalties by paying on time.",

"KEEP THIS TICKET!"

}

Dim lineSpacing As Integer = 18

For i As Integer = 0 To reminderLines.Length - 1

Dim fontToUse As Font = If(i = 1, fntBold, fntNote)

Dim lineX As Single = (bmp.Width - g.MeasureString(reminderLines(i), fontToUse).Width) / 2

g.DrawString(reminderLines(i), fontToUse, Brushes.Black, lineX, qrBottom + 10 + (i \* lineSpacing))

Next

End Using

Return bmp

End Function

' 🖨️ Print event

Private Sub PrintDoc\_PrintPage(sender As Object, e As PrintPageEventArgs) Handles PrintDoc.PrintPage

If TicketBitmap IsNot Nothing Then

e.Graphics.DrawImage(TicketBitmap, 0, 0)

End If

End Sub

**Team Composition**

**Name:** Joshua Nombrado Jaculba

**Address:** Bukid Area 4, Brgy. Sun Valley,

Paranaque City, Metro Manila

**Email Address**: joshuajaculba@gmail.com

**Contact Number**: (+63) 951-914-1629

1. **Personal Profile**

Year Level: Third Year

Course: Bachelor of Science in Computer Science

Team Designation: System Developer

**II. Educational Attainment**

Secondary :  Sun Valley National High School

                                        (2016-2023)

Primary            : Sun Valley Elementary School

   (2010-2016)



**Name:** John Paul Duran Navarro

**Address:** Dona Juana St. Brgy Kristong Hari

Quezon City, Metro Manila

**Email Address**: navarrojohnpaul315@gmail.com

**Contact Number**: (+63) 927-234-7468

1. **Personal Profile**

Year Level: Third Year

Course: Bachelor of Science in Computer Science

Team Designation: System Analyst

**II. Educational Attainment**

Secondary :  Urbiztondo Catholic School

                                        (2017-2023)

Primary            : Mangatarem I Central School

   (2012-2017)