PARKING MANAGEMENT SYSTEM FOR SM MALL OF ASIA

A Technical Documentation Presented to the Faculty

of Datamex College of Saint Adeline Inc.

In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science in Information Technology

By:

Mallory Makyla P. Chua

Ben O. Onde

Sean Roscoe D. Panday

Kim Razel Torrazo

INTRODUCTION

This section presents the Parking Management System and outlines the purpose and overview of this technical documentation as well as its scope. It gives the basis of how the system came to be developed and how developers, testers, administrators, and other stakeholders should utilize this document.

The Parking Management System (PMS) was created to streamline and automate the parking activities at SM Mall of Asia to eliminate the inefficiency of the manual processes. Currently, attendants enter vehicle details on paper tickets, a process that normally causes delays, lost or broken documents and long queues that adversely affect customer satisfaction. The PMS will address these issues by offering an online platform where cars can be registered when entering, payments can be generated automatically depending on the duration of stay, as well as the availability of parking slots which can be checked online. The system to them is represented by the administrators as a centralized dashboard through which they can keep track of operations in addition to managing user accounts and preparing an overview-report based on which they can enhance their decisions. The history of gaming has been an unforgettable one, which has been characterized by an evolution of graphic pixels to engulfing virtual reality and groundbreaking gameplay. With this technological innovation, the importance of games in society has also increased and pertained to all spheres of human life and growth.

Visual Basic.NET was utilized to create the interface and application logic of the system; the database management system was SQL Server. It is in local area network of the mall, hence it operates continuously, even when there is no internet connection. PMS enhances the integrity and security of data through role-based authentication, password encryption, and tracking system activities through access logs. In addition to ensuring that it could meet the existing operational needs, the PMS was to be scalable in future and could be extended by adding RFID, license plate recognition, and support of mobile applications. It is a documentation of system requirements, system design, installation and maintenance procedures in such a way that the system can sustain as a sound, cost effective and convenient solution to one of the biggest malls in the country.

System Architecture

Car Parking Management System uses a stand-alone client-server network that is intended to work within the local network of SM Mall of Asia. The Visual Basic.NET application is a presentation/business logic application with a direct interface to a SQL Server database using parameterized queries served by SQL Server Management Studio (SSMS). The system is reliable and secure as it eliminates the dependence on servers or internet connectivity, leading to uninterrupted operation of the system throughout the business hours of the malls.

The structure is divided into three layers. The Presentation Layer will include user-friendly attendants and administrator intuitive user forms. It offers modules that include the login, dashboards, vehicle registration, slot monitoring, transaction handling, as well as report generation modules. Business Logic Layer also captures the business processes involved in the parking process such as checking the input information, computing the duration and fees of parking, role-based access control and tracking slot availability. Lastly, Data Access Layer takes care of all communications between itself and the SQL Server database, issue structured queries, maintain the transactional consistency of the vehicle record, the slot allocation, a user account, and financial transaction.

High-Level Components and Their Interactions

Authentication, Vehicle Registration, Slot Management, Fee Calculation, Transaction Logging, Dashboard Reporting, User Administration are the various modules of the system. These modules combine to work seamlessly together to ensure seamless operations. For example, when an attendant registers the vehicle, it will validation check the input data, save the record to the database, will give an available parking spot and the dashboard solves the change in real-time. When the same vehicle leaves, according to the reminder time, the system calculates the total payment fee based on the inner and outer times, and records the payment transaction credentials, and releases the relevant parking position. The data generated can then be used to create summary reports that help with operational decision-making and improving accountability.

Deployment Architecture

The PMS is installed on selected Windows client machines in the mall, on which the Visual Basic.NET application is installed. These client computers contact a SQL Server data server running on the same local area network. To avoid interruptions, the deployment environment has uninterruptible power supply (UPS) devices to protect from power interruptions as well as backup storage devices for storing critical information. Since the system can only be accessed from within the network of the mall, the system does not have a lot of exposure to external threats which assures security and reliability.

INSTALLATION GUIDE

In this part, we will discuss the installation process of the PMS, including system requirements, installation steps, and configuration options. It helps to ensure that the system can be implemented in a standardized and reliable way.

System Requirements

Both client and server environment are needed for the system.

Component	Minimum Specification
Operating System	Windows 10 or later
Framework	.NET Framework 4.x with Visual Basic.NET support
Processor	Intel i5 Dual-core or equivalent
Memory	4 GB RAM
Storage	2 GB free disk space
Network	LAN connection within mall intranet

Table 1. Client Requirements

Component	Minimum Specification
Operating System	Windows Server 2016 or later
Framework	Microsoft SQL Server with SQL Server Management Studio (SSMS)
Processor	Intel i5/i7 Quad-core or equivalent
Memory	8 GB RAM
Storage	20 GB free disk space (for database and backups)
Network	LAN connection with UPS support to prevent downtime

Table 2. Server Requirements

Software Tool	Purpose
Visual Basic.NET	Development of user interface and application logic
SQL Server (SSMS)	Database creation, storage, and management
Git Version Control	Source code management and version tracking
Microsoft Word	Documentation and reporting support

Table 3. Software Dependencies

Installation Steps

1. Make sure that the client and server computers meet the minimum hardware/software requirements.
2. Install Visual Basic .NET + .NET Framework 4.x on all client machine(s).
3. Install SQL Server and SQL Server Management Studio (SSMS) on the server machine associated with your deployment.
4. Create the PMS database by running the given SQL scripts inside SSMS.
5. Initialize the Vehicles, Parking Slots, Users, and Transactions tables.
6. Provide startup data like parking slot data and administrator accounts.
7. Install PMS Visual Basic.NET application onto all client machines.
8. Database connection string to SQL Server instance on the network should be set up in your application.
9. Monitor client application to database server connectivity.
10. Conduct test execution for vehicle registration, slot monitoring and fee calculation to ensure system correctness.

Configuration Settings

1. Define the user role of the system: Attendant (limited access) & Administrator (full access).
2. Set passwords for all users, ensuring that they are strong.
3. Set the cost of parking and how it is calculated via application settings.
4. Define identifiers in the DB for slot representation for available and occupied parking.
5. Schedule database backup in SSMS and save backups in external drive or secure storage.
6. Add logging and tracking for failed logs, invalid inputs, and system errors
7. Limit the system only to the local area network (LAN) of the mall for security purposes.

CONFIGURATION GUIDE

The configuration stage for this software is one of the most important steps to ensure that the software is correctly configured to the operational policy of SM Mall of Asia. After installation, the administrator will need to set up the database, system parameters, user roles, and backup procedures in order to make the system ready for use every day.

Detailed Instructions for Configuring the Software

Database first needs to be created in SQL Server Management Studio (SSMS) The administrator sets up the PMS database and runs supplied SQL scripts to create the tables for Vehicle, Parking Slot, Transaction and User. Initial information such as parking slot identifiers and administrator credentials are then input into the database in order for the system to begin functioning. After the test, the application's configuration file should be modified to house the correct database connection string in order to facilitate communications between the Visual Basic.NET application and the SQL Server database. User roles will then need to be defined. Role based access is enforced as the system expects the administrator to set-up accounts for attendees as well as administrators. This can be achieved by granting the attendees limited access (e.g. only vehicle registration, fee calculation, and slot updates), while administrators are granted full access, including the ability to manage users, monitor transactions, and generate reports. Each account should have different login ID, and passwords are stored in encrypted form to ensure security of user data.

User Roles

The Car Parking Management System incorporates access control: role-based access control to manage users and their permission to carry out the corresponding functions in relation to their responsibilities. Along with securing sensitive information, this design also improves accountability by clearly dividing the responsibilities of attendants and administrators. Each user role has certain rights along with restrictions in using the system.

Attendant

- Registers vehicles entering the parking lot by taking plate numbers, time of entry, and designating parking spots.
- Performs calculations and processing of parking fees when departure is made from the facility.
- Real-time updates for slot state (including availability).
- No access granted that does not deal with the performance of daily operations, or allow the use of the full reporting features, and change environment settings.

Administrator

- Manages User Accounts (create, update or inactivate staff accounts managed by user account).
- Produces summary reports of parking activities at a daily, weekly or monthly level.
- Tracks system logs, slots usage and transaction logs via administrator dashboard.
- Defines system parameters, including car parking rates, fee calculation rules and slot number identifiers.
- Has full privileges to supervise attendants and oversee operations.

API DOCUMENTATION

This section introduces the API functionality of Car Parking Management System (PMS). It documents whether or not the system currently makes APIs available, what integration points may be available for future development, and what standards and data formats may be used.

List of APIs Exposed by the System

Now the Car Parking Management System is like a VB.Net standalone desktop application connected to an SQL Server database on the mall's local area network. Also, it does not expose public APIs or integrate with third-party services. All fundamental processes - from authentication, to vehicle registration, to slot tracking, fee calculation, transaction logging and report generating - are performed internally within the application itself. This model of communication provides ease of deployment and stability for offline operation, leaving limited possibilities of integration into systems outside the company.

Endpoint URLs, Request/Response and Parameters

Since the PMS is at the moment an internal system without external interfaces the APIs are expressed as logical interfaces between the client application and the database. These can be registered as modules for possible exposure in the future:

Authentication Module. Authenticates user credentials and manages role-based access control. Details like username and password are involved. Returns true or false with given role (attendant or administrator)

Vehicle Registration Module. Registers vehicle information including plate number, type, time of entry and allocated slot The reply acknowledges registration and changes slot status. Fee Calculation Module: Calculates exit data using vehicle ID and exit time to calculate duration and parking fee. The answer contains calculated charges, and updates transaction records.

Slot Availability Module. Used to obtain real time available slot by dashboard. The result is an array of slot identifiers with status (available or occupied).

Report Generation Module. Generates HV summary reports of parking activity for date ranges Parameters are provided (report type, time period, etc.) and totals for vehicles, fees, and usage statistics are returned.

Authentication and Authorization Requirements

Authentication and authorization are implemented internally by using role-based access. Attendees can self-register a vehicle, calculate fees and view slot availability, and administrators have complete control over user management, system properties, and report generation, to name just a few key features. User credentials are encrypted and all failed login attempts are audited. Additionally, if you plan to integrate with an external API in the future, you can add secure authentication mechanisms such as token-based authentication or OAuth 2.0 to ensure secure and controlled access thereafter.

DATABASE DOCUMENTATION

This section gives an overview of the design and structure of database for the Car Parking Management System. It includes the entity-relationship diagram (ERD), table descriptions and relationships, data migration and data backup that will keep information stored in a reliable and integrity-preserving way.

Entity-Relationship Diagram (ERD)

The PMS database is a simple relational database and comprises four main entities: Users, Vehicles, ParkingSlots, and Transactions.

- The Users table is used for system accounts - both administrators and attendants.
- The Vehicles table holds information about registered vehicles such as plate number, type, etc., and the slot number assigned to the vehicle.
- The status of the slots is tracked in the ParkingSlots table - which determines if the slot is open or taken.
- The Transactions table logs in all the entry and exit events with their timestamps, calculated duration and fees collected

Relationships

- Each Transaction has one Vehicle and a User (the staff person who completed the transaction).
- Each Vehicle will be allocated to one ParkingSlot at a time, and slot status information gets updated correspondingly.
- There is a one-to-many relationship between Users and Transactions; this relationship is meant to indicate accountability for the actions taken during a session logged in that User account.

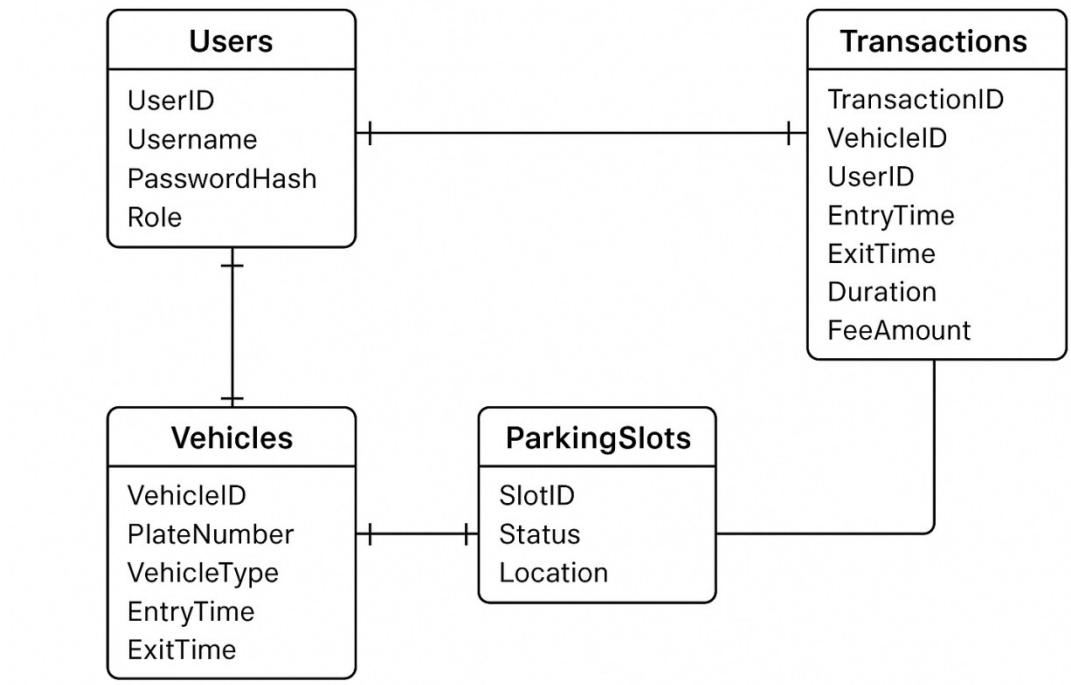

Figure 1. Entity-Relationship Diagram (ERD)

Description of Database Tables, Fields, and Relationships

1. **Users Table**

- Fields: UserID (PK), Username, PasswordHash, Role (Admin/Attendant)
- Manages authentication and authorization.

2. **Vehicles Table**

- Fields: VehicleID (PK), PlateNumber, VehicleType, EntryTime, ExitTime, SlotID (FK)
- Stores vehicle details and timestamps of entry and exit.

3. **ParkingSlots Table**

- Fields: SlotID (PK), Status (Available/Occupied), Location
- Tracks parking slot usage in real time.

4. **Transactions Table**

- Fields: TransactionID (PK), VehicleID (FK), UserID (FK), EntryTime, ExitTime, Duration, FeeAmount
- Logs transactions for fee calculation and reporting.

Relationships

- Users.UserID → Transactions.UserID (one user can process many transactions).
- Vehicles.VehicleID → Transactions.VehicleID (a vehicle can appear in multiple transactions over time).
- ParkingSlots.SlotID → Vehicles.SlotID (a slot may be assigned to different vehicles at different times).

USER MANUAL

This section deals with how to use the Car Parking Management System (PMS). It features step-by-step directions, navigational information, and explanations for common workflows. The user manual will allow for effective utilization of the system by staff in their daily operations, without the need for technical knowledge of the background.

User Login

1. Open the PMS application on the client computer for which you want to store the pass.
2. The Login Window will open asking for a User Name and Password.
3. Enter your required username and password.
4. Attendees access accounts with limited access to only operational functionality.
5. Accounts with extended access privileges are used by administrators.
6. Click the Login button.
7. If the credentials are correct, the related dashboard will be loaded. If this is wrong an error message will be displayed and the system will request re-entry.
8. Always sign out at the end of your shift using the Logout button in the main screen to lock your account.

Vehicle Registration (Entry)

1. From the attendant dashboard, click Register Vehicle.
2. Input the following details:
    - Plate Number
    - Vehicle Type
    - Select an available slot from the drop-down list.
3. The system automatically records the Entry Time.
4. Click Confirm Registration to save the record.
5. The system updates the Parking Slot status to Occupied and displays a confirmation message.

Vehicle Exit and Fee Calculation

1. From the dashboard, click Exit Vehicle.
2. Select the vehicle record either by entering the plate number or selecting from the active list.
3. The system automatically records the Exit Time.
4. The PMS calculates the parking Duration and corresponding Fee Amount based on the configured rates.
5. The attendant collects payment from the customer and confirms the transaction in the system.
6. The Parking Slot status automatically changes back to Available, and the transaction is logged in the database.

Slot Monitoring

1. Open the Slot Monitoring Panel from the dashboard.
2. The system displays a real-time list of all slots with their statuses:
    - Available – slot is free for use.
    - Occupied – slot is currently taken by a registered vehicle.
3. Administrators can view the overall parking capacity, while attendants use this panel to assign slots during registration.

Report Generation (Administrators Only)

1. Log in as an Administrator.
2. From the dashboard, click Reports.
3. Select the type of report:
    - Daily
    - Weekly
    - Monthly

4. Define the date range for the report if necessary.
5. Click Generate Report.
6. The system retrieves data from the Transactions table and produces a summary that includes:
    - Total number of vehicles processed.
    - Total fees collected.
    - Average duration of stay.
7. Reports may be viewed on-screen or exported for record keeping.

User Management (Administrators Only)

1. From the admin dashboard, click User Management.
2. To add a new user:
    - Click Add User, then enter username, password, and assign a role (Attendant or Administrator).
3. To update a user:
    - Select a user from the list, then modify details such as password or role.
4. To deactivate a user:
    - Select the account and click Deactivate. This prevents access without deleting the record.
5. Confirm all changes. The system updates the Users table in the database.

TROUBLESHOOTING GUIDE

This section provides an overview of troubleshooting the common problems that might arise while using the Parking Management System (PMS). It offers explanations of errors, possible troubleshooting actions and recovery options. The idea is to be able to allow staff and administrators to restore normal operations with minimal technical knowledge.

Common Issues and Error Messages

Invalid Login Credentials - This error occurs when a user enters an invalid username or password. The system returns an error message and does not give access.

Database Connection Error - This error occurs when the client program is unable to connect to the SQL Server database.

Slot Assignment Error - Occurs when an attendant tries to assign a slot which is already occupied.

Incorrect Fee Calculation - Can happen when there's a lack of entry or exit time or they are not entered accurately in the system.

Application Freezes/Crashes - Usually related to hardware constraints, power loss or incorrect installation.

Troubleshooting Steps and Resolutions

Invalid Login Credentials

1. Re-enter the correct username and password.
2. If forgotten, request a password reset from an administrator.
3. Administrators should verify the account exists and is active in the Users table.

Database Connection Error

1. Verify that the server computer hosting SQL Server is powered on.
2. Check the network connection (LAN cable or Wi-Fi intranet).
3. Confirm the application's configuration file has the correct connection string.
4. Restart the application after fixing connectivity issues.

Slot Assignment Error

1. Refresh the Slot Monitoring panel to confirm the slot status.
2. Assign another available slot.
3. If the problem persists, verify that the ParkingSlots table is updating correctly.

Fee Calculation Error

1. Ensure that the vehicle was registered properly at entry.
2. Check if the Exit Time was recorded in the transaction window.

Application Freezes/Crashes

1. Restart the client computer and reopen the application.
2. Check if the system meets minimum hardware requirements (RAM, storage, processor).
3. Ensure the server is connected to a UPS to prevent power interruptions
4. If frequent crashes occur, reinstall the application and reconfigure the database connection.

Contact for Technical Support

Role	Name	Email Address	Contact Number
Project Leader	Kim Razel Torrazo	torrazokim44@gmail.com	09218160315
Backend Engineer	Ben O. Onde	benonde0@gmail.com	09637690212
Frontend Designer	Mallory Makyla P. Chua	mclloryed.chua@gmail.com	09618480500
QA Specialist	Sean Roscoe Panday	pandaysean@gmail.com	09764394546

Table 4. Contact Information

CODE DOCUMENTATION

Below will be a brief account of the code structure, main modules and code standards in the Car Parking Management System (PMS). This is to ensure that the future developers, maintainers or other collaborators know how the codebase is organized and keep the development consistent.

TESTING DOCUMENTATION

This section gives a brief description of the code structure, major modules and the code standards of the Car Parking Management System (PMS). This is meant to make the future developers, maintainers or those who will work with it know how the codebase is organized and maintain a consistent development process.

Objectives and Strategies

Parking Management System testing stage was carried out to make sure that the software is used safely, reliably, and with ease. The main aim was to test the main user authentication and account management modules namely the Initial Setup, Login and Forgot Password features as these form the basis of all other functionality of the system.

The testing plan used a combination of the test in order to have a complete coverage. Black-box testing was used to test against the external behavior of the system, i.e. checking inputs and outputs to the functional requirements. White-box testing was carried out to test internal code logic, data processing and security code, such as validation of password hashing and excellent exception handling. Besides that, User Acceptance Testing (UAT) was conducted to ensure that the system fulfills the expectation of its targeted users regarding usability and clarity.

All testing was based on manual interaction with the system graphical user interface (GUI). Database validation, through SQL Server Management Studio (SSMS) was also done to ensure the reliability of results as sensitive data was saved in a safe location, and system responses matched anticipated results.

Testing Environment

This part reports the hardware, software, and test data that was used throughout the testing stage. It makes sure that testing was conducted in a controlled and well documented situations and results are reproducible and reliable.

Hardware Specification

Device	Processor	RAM	Operating System	Storage
Desktop PC	Intel Core i5-9400	8 GB	Windows 10 Pro	256 GB SSD

Table 5. Hardware Specification

Software Requirements

Category	Specification
Development Environment	Microsoft Visual Studio 2010 (VB.NET)
Database Server	Microsoft SQL Server 2019 Express
Database Admin Tool	Microsoft SQL Server Management Studio 18 (SSMS)
Framework	Microsoft .NET Framework 4.0

Table 6. Software Requirements

Test Data

To test properly and exhaustively a number of test data sets were prepared and used:

- **Predefined User Accounts -** Separate accounts that included Administrator, Staff, and so on were created to test the login and role-based access control.
- **Security questions -** Accounts were given their own security questions and answers to check the password recovery module.
- **Invalid Credentials -** The inability to verify the system regarding the error handling capabilities was checked by using incorrect usernames and inappropriate passwords.

- **Boundary Case Data -** Fields with empty contents and the maximum length of strings as the input were tested to confirm the stability of the form input processing.

Functional and Non-Functional Requirements

There were a set of test cases which were intended to test both non-functional and functional issues of the Parking Management System. In every case, the goal of the activity, the process, the desired solution and the actual solution were established to be traceable and accurate.

Functional Requirements

Test Case ID	Description	Expected Result	Actual Result	Status
TC001	Successful Login with valid credentials	Login successful; Main form opens	As expected	Pass
TC002	Failed Login with invalid password	Error message displayed, field cleared	Error displayed correctly	Fail
TC003	Initial System Setup	Account created; Login form displayed	As expected	Pass
TC004	Prevent Duplicate Username	Error message shown	As expected	Pass
TC005	Password Reset with correct answer	Password reset successful	As expected	Pass
TC006	Password Reset with incorrect answer	Security error displayed	As expected	Pass
TC007	Login Attempt with empty fields	Validation message displayed	As expected	Pass

Table 7. Functional Test Cases for User Authentication and Account Management

Non-Functional Requirements

Aspect	Findings
Security	Passwords and answers securely hashed before database storage.
Stability	Application handles exceptions (e.g., SQL errors) without crashing.
Usability	Interface clear and functional; feedback messages effective. Some improvements suggested (e.g., relabel "Account ID" to "Username", streamline password reset flow).

Table 8. Non-Functional Test Cases and Results

Testing Process

The Parking Management System testing was conducted in a sophisticated way by being able to test the outer behavior of the application, and the inner logic of the application. The integration of the various testing systems helped the team to ensure that the system was not just in a working situation as a user, this was an assurance that is also stable, secure and reliable at the code level. The main three methods were the Black-Box Testing, the White-Box Testing and the User Acceptance Testing (UAT).

Black-Box Testing

Black-box testing was aimed at checking the system in terms of how it would appear to an end-user. Under this method, the testers did not have to look into the real computer program, they have been able to engage with the application by inputting values and seeing the results. This approach was used to verify that the functionality of the system performed in line with the requirements and the messages when a check failed to be valid were correctly displayed.

In the case of Parking Management System, the black-box testing was implemented in regard to:

- Check successful login and failed login using achieveable and non-achievable credentials.
- Test that the Initial Setup form adequately dealt with and rejected invalid entries in the account creation.
- Confirm that system navigation between the Login, Set up and Forgot password was easy.

White-Box Testing

Whereas, black-box testing tested the system in the external environment, white-box testing tested the functionality of the system at the internal level. This method included performing a review of the source code, control structures and pay streams to ensure the logic was correct and that sensitive information was handled in a secure manner.

White-box testing in the Parking Management System was done to:

- Test the logic in the First Sets up (e.g., making sure that in the case of no users, the system allows setup by an administrator to be made).
- Make sure the HashPassword () procedure had always been carried out prior to the saving of credentials in the database.
- Reviewing exception handling Inspecting Try-Catch blocks, ensuring that error and troubles treated by SQL and otherwise managed gracefully and did not crash the application.
- White-box testing determined the reliability and security of the underlying structure by inspecting the system at the code level.

User Acceptance Testing (UAT)

The considered process had the final step as User Acceptance Testing, where its purpose is to ensure that the system addressed the expectations of its real users. In contrast to the other approaches, UAT attached importance to the usability and the practical functionality under real life set up.

In the case of the Parking Management System, UAT also implied running a virtual simulation of how an administrator would normally go about his work:

- The first account contents are created when the system is initially installed.
- Checking in by means of valid credentials and checking the simplicity of the log-in procedure.
- Security Questions questions can be used to recover access by creating a Forgot password feature.
- Giving feedback on the readability of instructions, the format of forms and experience.

Testing Type	Purpose	Application in the Parking Management System
Black-Box Testing	Focuses on checking the system's behavior from a user's point of view, without looking at the actual code. It ensures that the software responds correctly to different inputs and produces the expected outputs.	Used to confirm successful and failed login attempts, validate the Initial Setup form, and check the smooth navigation between Login, Setup, and Forgot Password screens.
White-Box Testing	Examines the system's internal structure and logic to verify that processes are working as intended and sensitive data is handled securely.	Applied to review the Initial Setup logic (If userCount = 0 Then), ensure that the HashPassword() function is always executed before saving credentials, and test exception handling with Try-Catch blocks.
User Acceptance Testing (UAT)	Conducted by end-users to confirm that the system is easy to use, meets business needs, and is ready for deployment in real-world conditions.	Performed by simulating an administrator's workflow: creating an account during first-time setup, logging in with valid credentials, using the Forgot Password feature, and providing feedback on clarity and usability.

Table 9. Comparative Overview of Testing Processes

Bug Tracking Issue and Log

In this section all bugs or problems identified during the test are recorded. The bugs are carefully controlled and logged with their identifiers, description, severity, reporter, current status and resolution. This hierarchical system of monitoring and ranking of problems assures the quick response of problems.

Bug ID	**Description**	**Severity**	**Status**	**Resolution**
B001	App crash if DB offline	Critical	Resolved	Added exception handling
B002	Login case-sensitive	Medium	Open	Query adjustment needed
B003	Password reset form stays open	Low	Resolved	Auto-close implemented
B004	Password fields not masked	High	Resolved	PasswordChar property set
B005	Cancel button misdirects user	Medium	Resolved	Corrected button logic

Table 10. Defect Reports

MAINTENANCE GUIDE

The maintenance plan of the Car Parking Management System (CPMS) offers a formal, systematized methodology to make the system reliable, secure and efficient throughout the lifecycle of the operations six. It establishes the tactical arrangements, measures and obligations of ensuring that the system is maintained and deal with anticipated as well as unanticipated problems.

Overall Maintenance Strategy

The general approach of sustaining the CPMS is a combination of proactive, corrective, and adaptive ones. Maintenance work is planned and maintained allowance throughout the processing to ensure that the system is running to it maximum capacity constantly. The major pillars of the plan are:

- **Proactive Monitoring:** The proactive monitoring is takes shape through following up system health, performance metrics and error logs and preventing a situation before it impacts the users.
- **Timely Problems Resolution:** Fast fix of bugs, error and performance issues to reduce downtimes and disruption of operations.
- **Technological Changes Adaptation:** Compatibility with Windows upgrades, SQL Server upgrades and other popular supporting technologies.
- **User-Centered Improvement:** Moving user feedback in such improvements to enhance user-friendliness, performance, and effectiveness.
- **Documentation and Training Updates:** To maintain errors free and work processes uninterrupted, keep the technical manuals intimate with the changes that occur in them, such as user guides and process notes.

Types of Maintenance

- **Corrective Maintenance:** It fixes faults like- login errors, crashing or transaction output errors.
- **Adaptive Maintenance:** Modifies the system to support the new releases of either SQL server, VB.Net or Windows operating system.

- **Perfective Maintenance:** Perfects the system to refine or optimize the system or make it workflow mundane.
- **Preventive maintenance:** Backs up, patches as well as checks logs in a bid to avoid maintenance lapses and failure.

Maintenance Schedule

Task	**Description**	**Frequency**	**Responsible Person**	**Status**
Database Backup	Produce and save system copies to avoid the destruction of data.	Weekly	Admin	Ongoing
Security Updates	Using security patches and updates, apply these to ensure data security.	Monthly	Development Team	Scheduled
Bug Fixes	Fix any errors reported to achieve a smooth functioning.	As needed	Support Team	Pending
Performance Check	Minimize response time and use of resources.	Quarterly	IT Team	Not Started

Table 11. Maintenance Schedule

Performance Monitoring

Metric	Description	Threshold	Monitoring Tool
Uptime	Measures the system's availability	≥ 99%	System Logs
Response Time	Time taken to load forms and reports	< 2 seconds	Manual Timer/Tools
Error Rate	Percentage of failed operations	< 1%	Error Logs

Table 12. Performance Monitoring Metrics

Version Control and Release Management Practices

In order to make sure that updates are carried out in an effective manner the following practices have been observed:

- **Version Control:** Every source code change is recorded in a repository, allowing the ability to roll back as well as documenting a history of development activity.
- **Release Management:** takes the form of Major. Minor.Patch (e.g. v123). Major releases are those releases that add new feature, minor releases are releases that update features and patches are releases that fix urgent problems.
- **Deployment:** New features, fixed bugs, and known weaknesses are announced by new releases every planned maintenance period; release notes are provided with each release.