Application Documentation

Introduction

This app is created for maintaining a database of customers, and bookings and to manage booking data in a business that organizes trips. Microsoft Access serves as the database back-end where the data were collected and stored, Microsoft Excel is the front-end while the VBA helps to connect and manipulate the data. The application contains functions for importing booking data, fetching customer specific bookings, and adding new bookings to the database. This is great for small to medium-sized businesses desiring an easy and effective solution.

Database

The back-end database is implemented in Microsoft Access and contains the following key tables:

- **Customers:** Holds information about customers, including unique IDs and personal details.
- **Bookings:** Contains records of bookings, each linked to a customer and a trip. Fields include BookingID, CustomerID, TripID, BookingDate, and Status.
- **Trips:** Describes available trips, with fields for trip details and unique identifiers.

Key Queries

1. **Import Bookings Query:** Retrieves all bookings with their associated customer and trip details. This is used in the ImportBookingsFromDB VBA procedure.
2. **Customer-Specific Query:** This query filters bookings by CustomerID for targeted information retrieval. It is utilized in the GetCustomerBookings VBA procedure.
3. **Validation Query:** Checks if a CustomerID exists before adding a booking, ensuring data integrity.

Front-End

The Excel interface is the primary front-end of the application. It includes the following sheets:

- **Importbooking:** Displays imported booking data with headers formatted for clarity. The sheet features color coding for different statuses:
    - Pending (Yellow)

- Shipped (Green)
- Delivered (Blue)
- Unknown (Red)

- **GetCustomerBooking:** Displays customer-specific bookings retrieved from the database. Includes columns for BookingID, TripID, and BookingDate.
- **Newbooking:** Allows the user to input data for a new booking. Includes fields for CustomerID, TripID, and BookingDate.
- **Pivot Table:** For sorting of customers based on different functionality.

The front-end design is user-centric, leveraging Excel's familiar interface to reduce the learning curve. Dropdowns, buttons, and formatted tables enhance usability.

VBA Middleware

The VBA code serves as middleware, connecting the Excel interface with the Access database. Below are brief descriptions of the primary subroutines:

1. ImportBookingsFromDB

- **Functionality:** Prompts the user to select an Access database file, retrieves booking data, and populates it in the Importbooking sheet.

Code:

```
' Prompt user to select the Access database file
With Application.FileDialog(msoFileDialogFilePicker)
    .Title = "Select Access Database"
    .Filters.Clear
    .Filters.Add "Access Database", "*.accdb;*.mdb", 1
    .AllowMultiSelect = False
    If .Show = -1 Then
        strPath = .SelectedItems(1) ' Get the selected file
    Else
        MsgBox "No file selected. Operation cancelled.", vbExclamation
        Exit Sub
```

```
    End If

        End With
```

The above VBA code was used to prompt users to pick a database specifically rather than attaching a specific path to the database. The database environment might change from time to time, but the code will still be effective, irrespective of the database environment.

- **Key Features:**
    - Automatically formats headers.
    - Applies color coding based on the Status field.
    - Uses CopyFromRecordset for efficient data transfer.

2. GetCustomerBookings

- **Functionality:** Retrieves bookings for a specific customer ID.
- **Key Features:**
    - Allows user input for CustomerID.
    - Filters records and populates the GetCustomerBooking sheet.
    - Includes error handling for invalid IDs or connection issues.

3. AddNewBookingToDB

- **Functionality:** Adds a new booking to the database.
- **Key Features:**
    - Validates CustomerID against the database to ensure it exists.
    - Formats the BookingDate for database compatibility.
    - Executes an SQL INSERT statement to add the booking.

Error Handling

The code employs robust error handling to manage issues like missing files, invalid input, and connection failures. User-friendly messages ensure clear communication of issues.

Conclusion

In conclusion, this application shows the powerful integration of Excel, Access, and VBA to efficiently manage business data, offering a strong foundation for streamlining operations and enhancing decision-making. Scaling the system to handle more processes and user evolving needs can be achieved by expanding the database to include additional tables and more complex relationships. Additionally, including additional front-end features, like interactive dashboards, automated reports, and user authentication can greatly increase usability by giving stakeholders real-time insights and secure access to important information. Deploying the solution on the cloud by hosting the Access database on a shared server or migrating to SQL Server ensures greater accessibility, data security, and performance optimization, making it suitable for multi-user environments and remote operations. This solution serves as a scalable and adaptable tool that can evolve alongside the business, driving efficiency and growth in data-driven decision-making.