Project Report: Shift-Level Inbound KPI Automation for Amazon Operations

Project Overview

Objective

The primary objective of this project was to address a significant gap in Amazon's operational data visibility. Previously, inbound Key Performance Indicators (KPIs) could only be viewed on a 24-hour basis, limiting the operations team’s ability to analyze data at the shift level. This project introduced a system that pulled data from multiple internal websites and consolidated it into shift-specific reports, enabling better decision-making, operational oversight, and performance monitoring.

Scope

Initially, the project was implemented at a single Amazon fulfillment site to test and validate the solution. Upon demonstrating its success, the project was expanded to encompass the regional level, benefiting multiple fulfillment centers across the region.

Project Background

Problem Statement

The inability to view inbound KPIs at the shift level posed several challenges for operations teams:

- **Delayed insights:** A 24-hour aggregation of data delayed corrective actions for underperforming shifts.
- **Limited granularity:** Lack of granular data made it difficult to pinpoint specific issues or opportunities tied to individual shifts.
- **Reduced accountability:** Teams working specific shifts had no immediate visibility into their performance metrics.

Proposed Solution

The proposed solution involved the creation of a suite of Excel VBA macros that would:

1. Pull shift-specific data from multiple Amazon internal websites.
2. Consolidate the data into structured, easy-to-read Excel sheets.
3. Enable seamless data retrieval and reporting for operations teams.

Project Implementation

Technical Approach

1. Data Retrieval

- **Web Scraping:** VBA macros were developed to pull data from Amazon's internal portals, including:
    - FCLM Portal for Process Path Rollup (PPR) reports.
    - RoboScout for site-specific metrics.
    - Adapt for learning curve and employee performance data.
- **Dynamic URL Construction:** Parameters such as warehouse ID, dates, and shift times were dynamically inserted into API calls.

2. Authentication

- Integrated Amazon's internal authentication system (Midway) to ensure secure and automated access to required datasets.
- A function was implemented to handle session cookies and prompt for re-authentication when necessary.

3. Data Processing

- Retrieved CSV files were processed directly in Excel, with macros:
    - Cleaning and formatting raw data.
    - Consolidating metrics into user-friendly worksheets.
    - Naming sheets dynamically based on site ID, date, and shift type.

4. Output and Reporting

- Generated shift-level KPI reports in Excel, including key metrics like units processed, time worked, and performance versus goals.
- Ensured outputs were clear, organized, and ready for use by the operations team.

Tools and Technologies

- **Microsoft Excel VBA:** Core scripting for data retrieval, processing, and reporting.
- **WinHTTP Library:** For making HTTP requests to internal websites.
- **Amazon Midway Authentication:** Secure access to internal APIs.
- **Amazon Internal Tools:** FCLM Portal, RoboScout, and Adapt APIs.

Code Breakdown by Macro

1. Sub `PPR_pull`

```
Sub PPR_pull()
    With Application
        Dim scrn As Boolean:        scrn = .ScreenUpdating
        Dim disp As Boolean:        disp = .DisplayAlerts
        Dim evnt As Boolean:        evnt = .EnableEvents
        Dim calc As XlCalculation:  calc = .Calculation
        .ScreenUpdating = False
        .DisplayAlerts = False
        .EnableEvents = False
        .CutCopyMode = False
        .Calculation = xlCalculationManual
    End With

    Dim ws As Worksheet, startDate As Date, endDate As Date, site$, proc$,
dest$

    ' Get user input from the "Control" sheet
    Dim siteStr As String, startDateStr As String, endDateStr As String,
startHourStr As String, endHourStr As String, startMinuteStr As String,
endMinuteStr As String, shiftType As String

    With ThisWorkbook.Sheets("Control")
        siteStr = .Range("B1").Value
        startDateStr = .Range("B2").Value
        endDateStr = .Range("B3").Value
        startHourStr = .Range("B4").Value
        endHourStr = .Range("B5").Value
        startMinuteStr = .Range("B6").Value
        endMinuteStr = .Range("B7").Value
        shiftType = .Range("B8").Value
        dayNumber = .Range("B9").Value
    End With

    site = UCase(siteStr)
    dest = "Site_" & site & "_Day_" & dayNumber & "_" & shiftType

    Dim url$: url = "https://internal-portal.com/reports/processPathRollup?"
    url = url & "reportFormat=CSV&warehouseId=" & site
    url = url & "&maxIntradayDays=1"
    url = url & "&spanType=Intraday"
    url = url & "&startDateIntraday=" & Format(startDateStr,
"yyyy%2Fmm%2Fdd")
    url = url & "&startHourIntraday=" & startHourStr
    url = url & "&startMinuteIntraday=" & startMinuteStr
    url = url & "&endDateIntraday=" & Format(endDateStr, "yyyy%2Fmm%2Fdd")
    url = url & "&endHourIntraday=" & endHourStr
    url = url & "&endMinuteIntraday=" & endMinuteStr

    Dim mwcookie As String
    mwcookie = MidwayCookie
```

```
    With CreateObject("WinHTTP.WinHTTPRequest.5.1")
          .SetAutoLogonPolicy 0
          .Open "GET", url, False
          .SetRequestHeader "Cookie", mwcookie
          .Send
    End With

    ' Save and process response
    Dim tempPath As String
    tempPath = Environ$("TEMP") & "\tempFclm.csv"
    Dim fso As Object: Set fso = CreateObject("Scripting.FileSystemObject")
    With fso.CreateTextFile(tempPath, True)
        .Write .ResponseText
        .Close
    End With

    With Workbooks.Open(tempPath)
        With .ActiveSheet
           ws.Range(ws.Cells(1, 1), ws.Cells(.UsedRange.Rows.Count,
.UsedRange.Columns.Count)).Value2 = .UsedRange.Value2
        End With
        .Close False
    End With

    Kill tempPath

End Sub
```

2. Sub `LC5_pull`

```
Sub LC5_pull()
    Dim sd As Date, ed As Date, fc$, ws$

    ' User input and settings
    Let ws = "Sheet1"
    Let fc = "LGA9"
    Let sd = #5/24/2024 7:00:00 AM#
    Let ed = #5/24/2024 5:30:00 PM#

    Dim url$: url = "https://adapt-iad.amazon.com/api/femida-
svc/GetRatePublishingReportV2?"
    url = url & "dateRangeType=CUSTOM&warehouseId=" & UCase(fc)
    url = url & "&reportStartTimeUtc=" & Format(sd, "yyyy-mm-dd\Thh:nn:ss") &
".000Z"
    url = url & "&reportEndTimeUtc=" & Format(ed, "yyyy-mm-dd\Thh:nn:ss") &
".000Z"

    Dim http As Object
    Set http = CreateObject("WinHTTP.WinHTTPRequest.5.1")
    http.Open "GET", url, False
    http.Send

    Dim csvPath As String: csvPath = Environ$("TEMP") & "\adaptExample.csv"
    Dim fso As Object: Set fso = CreateObject("Scripting.FileSystemObject")
    With fso.CreateTextFile(csvPath, True)
        .Write http.ResponseText
```

```
        .Close
    End With

    With Workbooks.Open(csvPath)
        With .ActiveSheet
            ThisWorkbook.Sheets(ws).Range("A1").Resize(.UsedRange.Rows.Count,
.UsedRange.Columns.Count).Value = .UsedRange.Value
        End With
        .Close False
    End With

    Kill csvPath
End Sub
```

3. Sub `NSTA_pull`

```
Sub NSTA_pull()
    Dim myURL As String, siteStr As String, startDateStr As String,
endDateStr As String

    ' Inputs
    With ThisWorkbook.Sheets("Control")
        siteStr = .Range("B1").Value
        startDateStr = Format(.Range("B2").Value, "yyyy-mm-dd")
        endDateStr = Format(.Range("B3").Value, "yyyy-mm-dd")
    End With

    myURL = "https://roboscout.amazon.com/view_plot_data/?" & _
             "sites=" & siteStr & _
             "&startDateTime=" & startDateStr & "T07:00:00" & _
             "&endDateTime=" & endDateStr & "T19:00:00"

    Dim http As Object
    Set http = CreateObject("WinHTTP.WinHTTPRequest.5.1")
    http.Open "GET", myURL, False
    http.Send

    Dim tempPath As String: tempPath = Environ$("TEMP") & "\nstaData.csv"
    Dim fso As Object: Set fso = CreateObject("Scripting.FileSystemObject")
    With fso.CreateTextFile(tempPath, True)
        .Write http.ResponseText
        .Close
    End With

    Dim newSheet As Worksheet
    Set newSheet = ThisWorkbook.Sheets.Add
    newSheet.Name = "NSTA_Report"

    With Workbooks.Open(tempPath)
        newSheet.Range("A1").Resize(.ActiveSheet.UsedRange.Rows.Count,
.ActiveSheet.UsedRange.Columns.Count).Value = .ActiveSheet.UsedRange.Value
        .Close False
    End With

    Kill tempPath
End Sub
```

4. Sub `OOWA_pull`

```
Sub OOWA_pull()
    Dim site As String, startDate As String, endDate As String, myURL As
String

    With ThisWorkbook.Sheets("Control")
        site = UCase(.Range("B1").Value)
        startDate = Format(.Range("B2").Value, "yyyy-mm-dd")
        endDate = Format(.Range("B3").Value, "yyyy-mm-dd")
    End With

    myURL = "https://roboscout.amazon.com/view_plot_data/?" & _
             "sites=" & site & _
             "&startDateTime=" & startDate & "T08:00:00" & _
             "&endDateTime=" & endDate & "T20:00:00"

    Dim http As Object
    Set http = CreateObject("WinHTTP.WinHTTPRequest.5.1")
    http.Open "GET", myURL, False
    http.Send

    Dim csvPath As String: csvPath = Environ$("TEMP") & "\oowaData.csv"
    Dim fso As Object: Set fso = CreateObject("Scripting.FileSystemObject")
    With fso.CreateTextFile(csvPath, True)
        .Write http.ResponseText
        .Close
    End With

    Dim ws As Worksheet
    Set ws = ThisWorkbook.Sheets.Add
    ws.Name = "OOWA_Report"

    With Workbooks.Open(csvPath)
        ws.Range("A1").Resize(.ActiveSheet.UsedRange.Rows.Count,
.ActiveSheet.UsedRange.Columns.Count).Value = .ActiveSheet.UsedRange.Value
        .Close False
    End With

    Kill csvPath
End Sub
```

Key Achievements

Operational Impact

1. **Improved Data Granularity:**
    - Enabled visibility into KPIs at the shift level instead of 24-hour aggregations.
    - Provided actionable insights to shift managers for better performance monitoring.
2. **Regional Expansion:**
    - After the successful pilot at a single site, the project was scaled to a regional level, benefiting multiple fulfillment centers.

3. **Enhanced Decision-Making:**
    - Operations teams could quickly identify underperforming shifts and implement corrective actions in real-time.
    - Shift-level accountability and performance tracking were significantly improved.

Efficiency Gains

- Reduced manual effort for data retrieval and reporting, saving time for managers and analysts.
- Automated processes that previously required several hours of manual work.

Adoption and Scalability

- The macros were designed with flexibility, allowing easy updates for new sites, metrics, or data sources.
- Clear documentation and user guides facilitated adoption by other fulfillment centers.

Challenges and Lessons Learned

Challenges

1. **Authentication Management:**
    - Handling Midway's cookie-based authentication was initially complex but was resolved through iterative testing.
2. **Dynamic Data Handling:**
    - Variations in data formats across different sites required robust error handling and data cleaning routines.
3. **Scalability:**
    - Ensuring the solution scaled seamlessly for regional use required modular design and optimization of macros.

Lessons Learned

- **Stakeholder Collaboration:** Frequent feedback from site managers ensured the solution addressed their specific needs.
- **Documentation:** Comprehensive documentation was critical for scaling the solution and training new users.
- **Testing:** Thorough testing across multiple sites ensured reliability and robustness.

Conclusion

This project successfully transformed how inbound KPIs are analyzed and utilized within Amazon operations. By enabling shift-level reporting, it empowered teams to make more informed decisions, driving improvements in efficiency and performance. The scalability and adaptability of the solution ensured its impact extended from a single site to a regional level, demonstrating its value across multiple fulfillment centers.