# **Automating Monthly Reports**

**Contents**

[**Automating Monthly Reports** 1](#_Toc181288187)

[Purpose 2](#_Toc181288188)

[Objectives 2](#_Toc181288189)

[Project Overview 2](#_Toc181288190)

[Background 2](#_Toc181288191)

[Stakeholders 3](#_Toc181288192)

[Approach 4](#_Toc181288193)

[Requirements 4](#_Toc181288194)

[Functional Requirements 4](#_Toc181288195)

[Non-Functional Requirements 4](#_Toc181288196)

[Software Development Lifecycle 4](#_Toc181288197)

[Planning 4](#_Toc181288198)

[Roles and Responsibilities 11](#_Toc181288199)

[Methodologies 11](#_Toc181288200)

[Design Patterns 11](#_Toc181288201)

[Implementation 11](#_Toc181288202)

[Tools and Technologies 11](#_Toc181288203)

[Code Structure 12](#_Toc181288204)

[Key Functions and Classes 12](#_Toc181288205)

[Code Examples 12](#_Toc181288206)

[Data Handling 12](#_Toc181288207)

[Testing 12](#_Toc181288208)

[Testing Frameworks 12](#_Toc181288209)

[Test Cases 12](#_Toc181288210)

[Error Handling 12](#_Toc181288211)

[Results and Analysis 12](#_Toc181288212)

[Outcomes 12](#_Toc181288213)

[Performance Metrics 12](#_Toc181288214)

[User Feedback 12](#_Toc181288215)

[Challenges and Solutions 12](#_Toc181288216)

[Challenges Faced 12](#_Toc181288217)

[Problem-Solving 12](#_Toc181288218)

[Conclusion 12](#_Toc181288219)

[Summary 12](#_Toc181288220)

[Future Improvements 12](#_Toc181288221)

Introduction

### Purpose

The purpose of this document is to provide a comprehensive overview of the project undertaken to automate the processing of monthly reports using various technologies. This project focuses on three specific reports: the EE Mobile Usage report, Teams Usage report, Teams Phone Usage Report, Windows Device Checks and iOS Device Checks. By automating these reports, the goal is to streamline the reporting process, reduce manual effort, and improve accuracy and efficiency.

### Objectives

The primary objectives of this project are:

**Efficiency:** To significantly reduce the time and effort required to generate and distribute monthly reports.

**Accuracy:** To minimise human errors associated with manual report generation.

**Consistency:** To ensure that reports are generated and distributed consistently every month.

**Scalability:** To create a solution that can be easily scaled to include additional reports or data sources in the future.

## Project Overview

### Background

#### EE Mobile Usage Report

The EE Mobile Usage Report details monthly mobile usage and costs incurred for international calls/ SMS and data roaming.

**Current Process:** Manually extracted from EE Mobile Manager in CSV format via scheduled data export. The data would then be processed manually using a pivot table to identify any users in the business with over £10 costs incurred for the month. Finally, the report would be manually delivered to the Finance Director via email for review.

**Objective:** Automate data cleansing, transforming, processing, and delivery to improve efficiency and accuracy.

#### Teams Usage & Teams Phone Usage Reports

The Microsoft Teams Usage Report details monthly Microsoft Teams usage including calls made, number of meetings attended etc.

**Current Process:** Manually extracted from Microsoft Teams Admin center in csv format. Manually split into two pivot tables: Standard Pivot (standard users) and ExCo Pivot (Executive Committee users). Manually delivered to the Finance Director via email for review.

**Objective:** Automate data extraction, processing, and delivery to streamline the workflow.

#### Finance Teams & Teams Phone Usage Report

The Finance Teams & Teams Phone Usage Report is essentially a slice of the Teams Usage and Teams Phone Usage Reports, specifically for the Accounts Team. It is delivered to Assura’s Property Finance Manager to review their team’s call statistics.

**Current Process:** Using a list of the current Accounting Team, data is extracted manually and placed onto a separate Excel Workbook, before being manually delivered to the Property Finance Manager for review.

**Objective:** Automate data extraction, processing, formatting and delivery to streamline the workflow.

#### InTune Device Checks (Windows & iOS)

The InTune Device Checks are used to identify anomalies such as out-of-date devices and devices with low storage in Assura PLC’s tenant.

**Current Process:** Extracted automatically from Microsoft Endpoint Manager in csv format, before the data would be manually cleaned/ transformed, formatted and then delivered to the IT Support Manager manually via email for review.

**Objective:** Automate data extraction, cleansing, processing, formatting and delivery to streamline the workflow.

### Stakeholders

The key stakeholders for the project include:

**Me:** Responsible for developing and maintaining the automation scripts and ensuring their smooth operation.

**Finance Director:** Receives the EE Mobile Usage Report, used to identify any costs incurred for the previous month.

**IT Support Manager:** Receives the Teams Usage Report, Teams Phone Usage Report, Windows Device Checks and iOS Device Checks.

**Property Finance Manager:** Receives the Finance Teams and Teams Phone Usage reports, used for decision-making.

### Approach

The project should follow a structured approach based on the Software Development Life Cycle (SDLC), ensuring thorough planning, design, implementation, testing, and maintenance of the automation scripts. In this document I will detail each phase of the project, providing insights into the methodologies and technologies used, as well as the challenges encountered and solutions implemented.

## Requirements

### Functional Requirements

* Automate data extraction using Application Programming Interfaces (APIs).
* Process and format the data into readable reports.
* Distribute the reports automatically via email.

### Non-Functional Requirements

* The project should ensure data security and privacy- no one other than the stakeholders mentioned should be able to gain access to the reports in question.
* The scripts should run within a specified time frame on the first day of each month.

## Software Development Lifecycle

### Planning

**Objective:** Here I defined the goals, scope, and requirements of the project.

**Tasks:**

* Identified the reports to be automated: EE Mobile Usage Report, Teams Usage Report, Teams Phone Usage Report, InTune Device Report.
* Determined the [functional](#_Functional_Requirements) and [non-functional](#_Non-Functional_Requirements) requirements.
* Identified [stakeholders](#_Stakeholders) and gathered their input to understand their needs and expectations.

**My Role:** As this was my project and something that I’d decided to do myself, I was responsible for defining the project scope, gathering requirements, and ensuring alignment with stakeholder expectations (reports formatted correctly and delivered in a timely manner).

#### 2. Requirements Analysis

**Objective:** Gather and analyse detailed requirements to ensure the scripts meet the needs of the project.

**Tasks:**

* Documented [functional requirements](#_Functional_Requirements) such as data extraction, report generation, and email distribution.
* Documented [non-functional requirements](#_Non-Functional_Requirements) including performance, security, and usability.
* Selected tools and technologies: Python, pandas, openpyxl, Microsoft Graph API.

#### 3. Design

##### Solution Architecture Design

Designing the solution architecture involves creating a high-level blueprint of how the different components of the system will interact and work together to achieve the project’s goals. This ensures that all parts of the system are well-organised, scalable, and maintainable.

###### High-Level Design

* **Components**:
  + **Data Extraction**: Scripts to extract data from EE Mobile Manager, Microsoft Teams, and Microsoft Endpoint Manager, using APIs.
  + **Data Processing**: Scripts to clean, transform, and format the extracted data.
  + **Report Generation**: Scripts to create the final reports in the required format.
  + **Email Distribution**: Scripts to automatically email the reports to the respective stakeholders.

**Interactions**:

* Data is extracted from the sources and passed to the data processing component.
* Processed data is then sent to the report generation component.
* Generated reports are passed to the email distribution component for delivery.

###### Detailed Design

* **Technologies and Tools**:
  + **Python Libraries**: pandas for data manipulation, openpyxl for Excel report generation, datetime module for manuplating dates and times, smtplib for sending emails.
  + **APIs**: Microsoft Graph API for extracting data from Microsoft Teams and InTune. Microsoft Graph API provides a unified programmability model that you can use to access data in Microsoft 365, Windows, and Enterprise Mobility + Security.
  + **Task Scheduler:** to run the scripts automatically on the first day of each month.
  + **GitHub:** code repository used to execute my scripts.

**Data Flow**:

* **Data Extraction**:
  + Use APIs to extract data from EE Mobile Manager, Microsoft Teams, and Microsoft Endpoint Manager.
  + Store the extracted data in temporary storage (e.g., CSV files or in-memory data structures).
* **Data Processing**:
  + Load the CSV files into pandas dataframes.
  + Clean and transform the data (e.g., filter out irrelevant data, handle missing values).
  + Apply business logic to identify key metrics (e.g., users with over £10 costs).
* **Report Generation**:
  + Use openpyxl to create Excel reports.
  + Format the reports with pivot tables and charts as needed.
* **Email Distribution**:
  + Use smtplib to send the generated reports to the stakeholders via email.

###### Data Flow Design

**Data Sources**:

* **EE Mobile Manager**: Extract monthly mobile usage data.
* **Microsoft Teams Admin Center**: Extract Teams usage data.
* **Microsoft Endpoint Manager**: Extract device data.

**Data Extraction**:

* Schedule data extraction scripts to run on the first day of each month.
* Use APIs to fetch the data and store it in CSV files.

**Data Processing**:

* Load the CSV files into pandas dataframes.
* Clean and transform the data (e.g., filter out irrelevant data, handle missing values).
* Apply business logic to identify key metrics.

**Report Generation**:

* Use openpyxl to create Excel reports.
* Format the reports with pivot tables and charts.

**Email Distribution**:

* Use smtplib to send the reports to the Finance Director, IT Support Manager, and Property Finance Manager.

##### Methodology

Due to the fact that this project lasted under a week, I did not fixate too much on the methodology used, however, I would say that I used the **Agile methodology** as it allows for iterative development and continuous feedback, which is ideal for a project that may require adjustments based on stakeholder feedback; for example, if a manager receives a report and they don’t like the formatting, or they find that it hasn’t been cleansed correctly as there is data missing due to errors in my script.

This is how the project could have looked using the Agile methodology:

###### Sprint Planning

Sprint 1: Data Extraction

**Goal:** Automate data extraction from EE Mobile Manager, Microsoft Teams, and Microsoft Endpoint Manager.

**Deliverable:** Scripts to extract data and store it in CSV files.

Sprint 2: Data Processing

**Goal:** Clean and transform the extracted data.

**Deliverable:** Scripts to process data using pandas.

Sprint 3: Report Generation

**Goal:** Generate formatted reports.

**Deliverable:** Scripts to create Excel reports using openpyxl.

Sprint 4: Email Distribution

**Goal:** Automate the distribution of reports via email.

**Deliverable:** Scripts to send reports using smtplib.

###### User Stories

EE Mobile Usage Report:

“As a Finance Director, I want to receive a monthly EE Mobile Usage Report so that I can review costs incurred for the previous month.”

Teams Usage Report:

“As an IT Support Manager, I want to receive a monthly Teams Usage Report so that I can monitor usage and identify any issues.”

Daily Stand-ups

Format: Each team member answers three questions:

What did you do yesterday?

What will you do today?

Are there any blockers?

###### Sprint Retrospectives

**Format:** Discuss what went well, what didn’t, and how to improve.

**Example:** “In the Sprint 1 retrospective, we identified that data extraction took longer than expected due to API rate limits. We decided to implement data caching to improve performance.”

#### Design Patterns

**Singleton Pattern:** Used to ensure a single instance of the configuration settings throughout the scripts.

**Factory Pattern:** Applied to create objects for different types of reports, promoting code reusability and scalability.

#### 4. Implementation

Objective: Write the code to implement the design.

##### EE Mobile Usage Report

import pandas as pd

from openpyxl import load\_workbook

from openpyxl.styles import numbers

# Correct the file path handling

input\_file\_path = r"C:\Users\Luke.Bryson\Documents\343200\_8129837\_20241010105259.csv" # Use raw string or double backslashes

output\_file\_path = r"C:\Users\Luke.Bryson\Documents\2024.09 EE Bill Breakdown.xlsx"  # Specify your desired output directory

try:

    df = pd.read\_excel(input\_file\_path)

except Exception as e:

    print(f"Error reading the Excel file: {e}")

    raise

# Remove '£' and convert Cost to float

try:

    df['Cost'] = df['Cost'].str.replace('£', '').astype(float)

except Exception as e:

    print(f"Error processing the Cost column: {e}")

    raise

# Create pivot table with the sum of costs for each user

try:

    pivot\_table = df.pivot\_table(index='User name', values='Cost', aggfunc='sum')

except Exception as e:

    print(f"Error creating pivot table: {e}")

    raise

# Filter users with a sum of costs equal to or exceeding £10

try:

    users\_above\_10 = pivot\_table[pivot\_table['Cost'] >= 10].index

except Exception as e:

    print(f"Error filtering users: {e}")

    raise

# Format costs as currency in the original DataFrame

try:

    df['Cost'] = df['Cost'].apply(lambda x: f"£{x:.2f}")

except Exception as e:

    print(f"Error formatting costs: {e}")

    raise

# Create an Excel writer object and save to file

try:

    with pd.ExcelWriter(output\_file\_path, engine='openpyxl') as writer:

        # Write the main DataFrame to the Excel file

        df.to\_excel(writer, sheet\_name='All Data', index=False)

        # Write the pivot table to the Excel file

        pivot\_table.to\_excel(writer, sheet\_name='Pivot Table')

        # Add filter to the header in the 'All Data' sheet

        workbook = writer.book

        worksheet\_all\_data = writer.sheets['All Data']

        worksheet\_all\_data.auto\_filter.ref = worksheet\_all\_data.dimensions

        # Write detailed breakdown for each user with a sum of costs equal to or exceeding £10

        for user in users\_above\_10:

            user\_df = df[df['User name'] == user]

            user\_df.to\_excel(writer, sheet\_name=user, index=False)

            # Apply filter and sort to each user's sheet

            worksheet\_user = writer.sheets[user]

            worksheet\_user.auto\_filter.ref = worksheet\_user.dimensions

            worksheet\_user.auto\_filter.add\_sort\_condition(f"K2:K{worksheet\_user.max\_row}")

except Exception as e:

    print(f"Error writing to Excel: {e}")

    raise

# Re-open the workbook to apply additional formatting

try:

    workbook = load\_workbook(output\_file\_path)

    # Apply formatting and sorting to each sheet

    for sheet\_name in workbook.sheetnames:

        worksheet = workbook[sheet\_name]

        # Format the 'Cost' column as currency

        for row in worksheet.iter\_rows(min\_row=2, min\_col=worksheet.min\_column, max\_col=worksheet.max\_column):

            for cell in row:

                if cell.column\_letter == 'L' or cell.column\_letter == 'B':  # Adjust column letter based on your data

                    cell.number\_format = '£#,##0.00'

        # Apply filter and sort by 'Cost' column

        worksheet.auto\_filter.ref = worksheet.dimensions

        cost\_col\_letter = 'L' if sheet\_name in users\_above\_10 else 'B'  # Adjust column letter based on your data

        worksheet.auto\_filter.add\_sort\_condition(f"{cost\_col\_letter}2:{cost\_col\_letter}{worksheet.max\_row}")

    # Save the workbook again after re-opening

    workbook.save(output\_file\_path)

except Exception as e:

    print(f"Error applying additional formatting: {e}")

    raise

print(f"Excel file created successfully at {output\_file\_path}!")

How did I implement error handling and logging to ensure robustness?

#### 5. Testing

Objective: Ensure the scripts work as expected and meet the requirements.

Tasks:

Conducted unit testing to test individual functions and components.

Performed integration testing to test the interaction between different components.

Carried out user acceptance testing to get feedback from stakeholders.

Test cases

#### 6. Deployment

Objective: Deploy the scripts to the production environment.

Tasks:

Set up the production environment (GitHub).

Deployed the scripts and ensured they were running correctly.

Provided documentation and training for users.

#### 7. Maintenance

Objective: Maintain and update the scripts as needed.

Tasks:

Monitored the scripts for any issues.

Made updates and improvements based on user feedback.

Ensured the scripts remained compatible with any changes in the data sources or requirements.

## 

## Results and Analysis

### Outcomes

### Performance Metrics

### User Feedback

## Challenges and Solutions

### Challenges Faced

### Problem-Solving

## Conclusion

### Summary

### Future Improvements

Automate sign-off using PowerAutomate/ Planner/ Approvals?