**Bachelor´s Thesis**

For the degree programme Business and Engineering

**Time-Efficient KPI Automation with Customizable PowerPoint Export Functionality**

In cooperation with

Technical University of Applied Sciences Würzburg Schweinfurt

Author: Gilbert Musinguzi

Matriculation no.: 3821036

Submission date:

1st Examiner: Prof. Schwindl-Braun, Kurt

2nd Examiner:

Contents

[**Introduction** 3](#_Toc201581834)

[**Why this research** 3](#_Toc201581835)

[**Economic Benefits** 4](#_Toc201581836)

[**Research Question** 4](#_Toc201581837)

[**Objectives of the research** 5](#_Toc201581838)

[**Scope and Limitations** 6](#_Toc201581839)

[**Why This scope matters / considered.** 7](#_Toc201581840)

[**Literature Review** 8](#_Toc201581841)

[**Existing tools for KPI Generation** 8](#_Toc201581842)

[**Python’s Role in Data Automation** 8](#_Toc201581843)

[**Data Quality and Processing challenges** 9](#_Toc201581844)

[**Use cases of KPI automation** 11](#_Toc201581845)

[**Demand for KPI Automated reports** 12](#_Toc201581846)

[**Methodology** 13](#_Toc201581847)

[**System Architecture Overview** 13](#_Toc201581848)

[**KPI Definition and Formation** 14](#_Toc201581849)

[**Processing data with python** 16](#_Toc201581850)

[**Summary** 40](#_Toc201581851)

[Bibliography 42](#_Toc201581852)

# **Introduction**

Key performance indicators are important metrics that help businesses get a summary of data and understand how their business are performing i.e., they are used for management and control of companies [1]. The main importance of key performance indicators is allowing managers make actionable decisions, gauge progress and insights that make people across the organisation make decisions [2]. This paper discusses how different organisations can leverage python to automate KPI generation to improve on performance by reducing time taken to manually compute KPIs [3]. It further illustrates the procedures on how python can be used to automate KPIs generation reducing time and improving on efficiency and accuracy [4].

## **Why this research**

KPIs have proven to be important when it comes to decision making because it allows managers to make actionable decisions based on the existing data to make rational decisions [5]. As the world develops, a growing need for autonomy in many sectors rises and one of them is in business intelligence with autonomous data analytics being the driver of business analytics in the future as it will be integrated into our lives [6].

Moreso, majority of KPI reports are generated manually which has proven to be redundant and not time efficient which can be misguiding when making decisions amongst stake holders [7]. Manual KPI generation poses a few challenges such as time lost during data collection, accuracy since data needs to be cleaned and formatted before it accurately analysed, the demand for real time data is also on a rise and sometimes manually generating real time data has proven to be ineffective because it requires a lot of time to compute [8].

Furthermore, limitations from software programmes like MS excel which are used by approximately 54% of company’s world wide [9] have bottle necks such as data type limits, limitations in statistical analysis and performance limitations among others can result into inaccurate KPIs generated [10].

Lastly, this paper aims at offering a guideline on how different organisations can adopt the existing solutions for business automation specifically with python and optimize their KPI generation and improve of operational efficiency [11].

## **Economic Benefits**

Beyond efficiency gains, KPI automation delivers measurable business insights used to monitor the progress of business operations and identify opportunities among others. McKinsey Research (2022) estimates that companies automating their performance reporting save an average $250,000–$1 million annually by reducing labour costs and minimizing errors that lead to poor decision making [12]. Moreso, standardized KPI reports ensure consistency across different use cases, eliminating anomalies that often arise from manual reporting.

Furthermore, Python-based KPI automation is important because of its versatility [13]. Python is an open-source software that allows different users to leverage different libraries such python-pptx [14], pandas among others to automate report generation [15]. Python scripts can process large datasets efficiently and accurately reducing on the time spent collecting and analysing data in small quantities due MS excel limitations. This scalability is essential for developing and growing organisations that must continuously expand their analytics capabilities without overhauling existing systems [16].

Moreover, automated KPIs are crucial for businesses as it streamlines data collection, analysis and reporting, freeing up resources for more strategic activities. This automation improves data accuracy, report formatting and consistency, allows better decision making and ultimately leads to cost saving since it eliminates redundant tasks that often times have been proven to be time consuming [17]. By automating KPI report generation this allows businesses/ organisations to adapt and thrive in a dynamic business environment.

Lastly, automated KPI generation allows access to real time metrics and underlying reasons for changes which facilitates proactive decision making [18]. Data driven insights also help organisations and businesses identify areas of improvement which may include markets, customers, suppliers among others [19].

## **Research Question**

Business analysts, project managers, marketers among others spend on average 24% of their time on data collection and management [20]. This results in excessive time extracting and computing KPIs from raw data, designing presentation slides for stakeholders and ensuring consistency and accuracy in reports. The solution would be to streamline these processes by leveraging different software such as python and SQL to minimize repetitive tasks and enabling faster decision making and eventually increase opportunities for growth [21].

Given the opportunities and challenges, the research question arises “**How organisations can strategically optimize their operations by reducing time spent to manually develop KPI reports by leveraging python and its Inbuilt libraries.”**

Organisations will be able to create and customize ad hoc dashboards for different use cases allowing managers and employees apply necessary filters and generate/download customizable power point reports that allow them track organisational progress identify upcoming trends, present findings to different stake holders and improve relationships by communicating effectively. Lastly, this paper aims to illustrate how time in generating manual power point reports can be reduced by leveraging software such as python and SQL to streamline and automate processes.

## **Objectives of the research**

**Objective 1.** The objective of this research is to develop a comprehensive framework that enables organisations to automate their KPI report process across various business functions. Setting these guidelines and implementation methodologies will empower enterprises to systematically transform their manual KPI tracking systems into automated, python-based solutions. The proposed frame work will be designed with sufficient flexibility to enable users select KPIs applicable to their use cases.

**Objective 2.** Reduce time spent by organisations when manually generating power point KPI reports by leveraging python and SQL [22]. Traditional manual process often involving data extraction from multiple data sources and manual PowerPoint compilation are not only labour intensive but also time consuming and also prone to human error. This research aims to streamline the KPI generating process and reduce redundancy resulting into more time efficient reports generated and effective decision making done.

**Objective 3.** Allow organisations and business download customizable PowerPoint reports by eliminating manual design work while maintaining brand consistency and data accuracy. This allows different stake holders / users apply filters such as dates to specify the time that may be of key interest to them [23].

The key components for this:

1. Template – Based Automation

Develop a variety of different templates using PowerPoint Presentation for different use cases.

Data visualization charts using (Matplotlib)

1. One click Export system

Python backend leveraging different libraries like python-pptx to generate downloadable .pptx files**.**

## **Scope and Limitations**

**Scope.** This research paper aims at covering how companies that rely on structured data to make decisions can automated their KPI reports that are generated. Structured data is information organized in a predefined format, making it easily accessible and analysable by both humans and computers [24].

Example of structured data include [25];

* excel spreadsheet
* relational data bases
* CRM systems
* inventory control systems

Given the prevalence of structured data in business operations, this research will focus on automating KPI extraction, computation and visualization from such sources, with a particular emphasis on Excel – based data as the primary use case.

The study will:

1. Develop a python-based automation framework that;

* Reads structured data.
* Cleans, transforms and computes KPIs using libraries like numpy and pandas.
* Generates structured Power Point reports.

1. Handle challenges in manual KPI generation such as scalability constraints (handling very large data efficiently) [26].

**Limitations.** While this paper will provide a robust framework of python that handles large data sets, it will not cover the following:

* Unstructured / semi-structured data and these include emails, social media posts, PDF reports among others [27].
* Real time data. This frame is highly dependent on how frequent data bases are updated for new data to be processed.

Therefore, concentrating on structured data particularly Excel, this study delivers a foundational yet scalable approach that organizations can extend to other structured data sources to optimize their KPI report generation.

### **Why This scope matters / considered.**

Excel/SQL are ubiquitous data sources. It is estimated that over 750 million users rely on spreadsheets for data management [28] more so, it is estimated that 7 million people are using SQL world-wide making it critical them critical as a starting point [29]. Moreso, structured data represents over 80% of enterprise decision making inputs offering three critical advantages for initial automation efforts [30].

Firstly, its nature makes it easy for data to be read and converted into different forms which aligns with python processing strengths [31]. Secondly, the ROI (Return on Investment) is measurable and immediate automating structured data reporting typically results into 70% time saving versus manual methods, compared to the 50% efficiency gains when tackling unstructured data like emails or PDFs [32].

Lastly considering the rate at which data is growing annually paired with large use of CRM tools such as excel this scope is both timely and essential for enabling organizations to efficiently manage their expanding data volumes while maintaining accurate, real time KPI reporting. Furthermore, the world currently has approximately 149 zettabytes of data with 80% of it being unstructured. Structured consist of 20% and accounts for most rational decisions made in organisations [33].

# **Literature Review**

The concept of Key performance Indicators evolved from various sources. The first practices in performance measurement dates back to the 20th century. Moreso, the history of manual KPI tracking is a foundational method for performance monitoring, spans from the industrial revolution to the digital age [34]. Initially, manual tracking relied on paper – based systems and spreadsheets, evolving through database management tools and eventually integrated software solutions.

In today’s day and age where the world is gravitating more towards autonomy and artificial intelligence manual systems / methods have limitations, including potential for errors, time lagged insights among others. In today’s fast-paced business world, data is everywhere. From website clicks to customer purchases to supply chain metrics, business is drowning in information. The crucial value is turning this data into insights that drive decisions and spark action. With this business in today’s day and edge are leveraging AI and business autonomy to transform the field of business intelligence as we know it [35].

For many years, the core principle of business intelligence has been about collecting data, generating reports, and hoping that someone makes the right decision. As the world develops and data increases on a daily basis, there are constraints that limit data analysis in the world of business intelligence. But if analytics could do more than just generate reports stake holders could actively respond to changes, commend actions and even make decisions automatically with little to no human intervention required [36].

## **Key benefits of autonomy in business intelligence**

1. Automated data analysis.

Manual data analysis often required significant time and effort to process and analyse and is prone to human errors. Leveraging python automates the process, enabling business to quickly sift through large data sets and find meaningful patterns.

**For example:**

**Retail**. Analysis of customer purchase histories to identify which products are frequently bought together. This helps businesses create effective strategies and capitalize on opportunities.

**Healthcare**. Hospitals can rely on autonomy to analyse patient data, uncover patterns in diagnoses, and recommend treatments more efficiently than manual reviews.

By automating repetitive tasks, businesses free up time for their teams to focus on strategic initiatives rather than processing data manually which becomes redundant in the long term [37].

## **Existing tools for KPI Generation**

The tools used for manual KPI generation mainly comprises of a mix of commercial business intelligence (BI) platforms and open-source solutions, each with the distinct advantages and limitations. Tools such as Power Bi, Tableau offer user-friendly interfaces and excellent visualization capabilities making them highly applicable to different enterprises however users usually have manually design PowerPoint reports to summarize what is displayed by dashboards in a company [38].

Python has emerged with powerful middle ground leveraging libraries like python pptx to bridge the gap between affordability and customization, enabling automated KPI report generation [39]. Despite these options, many organizations rely on manual Excel – based processes, highlighting a persistent gap in the accessible, end to end automation solutions that balance ease of use good functionality [40].

## **Python’s Role in Data Automation**

Python has become crucial for automation of data in the modern age due to its versatility, extensive library options and ease in integration [41]. Unlike proprietary tools with limited framework, python enables automation from data extraction to data aggregation and graphical visualization of data with no human intervention [42]. Python leverages libraries like NumPy, Plotly and matplotlib among others to streamline redundant tasks that can be time consuming.

Pythons’ compatibility with APIs, cloud platforms enable descriptive KPI reporting [43]. This flexibility makes python particularly valuable for organizations seeking cost effective solutions to automate their manual KPI report generation that proprietary Bi tools cannot provide.

Python is a highly valued in automation due its simplicity, readability and extensive libraries and versatile capabilities. With these characteristics, it makes python one of the best open-source software’s for automating tasks easier and faster, leading to increased efficiency and production [44].

Another key role of python in data automation is advanced analytics is possible as a result of python libraries such as NumPy, pandas combined with visualization tools like Matplotlib and seaborn, python can also automate the generation of insights and reports making it one stop solution for end-to-end data automation [45]. This active community and development ensure that python remains at the forefront of innovation in data automation, adapting to emerging technologies and industry demands [46].

Furthermore, cross platform compatibility makes python a versatile language that operates across various operating systems without modification, making it suitable for automating tasks across different environments [47]. One of its key strengths is its ability to integrate seamlessly with various databases, file formats, and APIs. Libraries like Psycopg2 allow Python to interact with SQL databases such as PostgreSQL providing a suitable framework to automate data extraction and transformation across different sources [48].

## **Data Quality and Processing challenges**

Data quality and processing present significant challenges in KPI automation, as unreliable input data directly comprises of performance metrics. Common issues include missing values, inconsistent formatting e.g., mixed date styles, duplicate records, and anomalous outliers all of which affect the aggregated results which distort KPIs if left unaddressed [49].

Python’s ecosystem provides powerful solutions leveraging libraries such as pandas that handles missing data and any anomalies with regards to numerical discrepancies. For text-based inconsistencies machine algorithms are used to consolidate anomalies in data. The consequences of poor preprocessing are severe as studies show that 34% of automated KPI errors stem from inadequate data cleaning, resulting into wrong decision making that have been fatal to many companies [50]. In regulated industries such as finance and health care, preprocessing must also address compliance through anonymization techniques and audit trails [51].

**Challenges associated when data preprocessing is being executed include the following** [52]**.**

**Data Quality Challenges** [53]**.**

1. **Inaccurate Data.** Errors in data entry, outdated data sources, and inconsistences across various data systems can lead to inaccurate results and poor decisions that can be costly.
2. **Incomplete data.** Missing data points or records can prevent a comprehensive understanding of situation and affect accuracy of analyses.
3. **Inconsistent data.** Discrepancies in data formats such as units can lead to wrong aggregated results which can falsely affect decision making.
4. **Duplicate data.** The presence of duplicate or repeated records can lead to inaccurate analysis, wasted resources and confusion.

**Data Processing challenges** [54]**.**

1. Handling Large volumes of data. Managing large datasets, including “big data”, requires robust processing infrastructure and algorithms. Data preprocessing transforms data into a format that’s more easily effectively and processed. Several tools and methods are used in data preprocessing and they include the following.

* Sampling. This approach selects a representative subset from a large population.
* Transformation. This is a way to manipulate raw data to produce a single input.
* Denoising. This removes noise from data or any outliers.
* Imputation. This method synthesizes statistically relevant data for missing values.
* Normalization. A way of organizing data for more efficient access.
* Feature extraction. This approach pulls out a relevant feature subset that’s significant in a particular context.

1. Integrating Diverse Data sources. Combining data from various sources with different formats, structures, and technologies can be complex.
2. Scalability. Ensuring that data processing systems are robust enough to handle ever growing data is essential for optimal workloads without compromising performance. It is important to ensure that computers housing the framework of processing are robust enough to handle growing data.
3. Real-Time Processing. Processing data in real-time to support timely decision-making and operational needs can be challenging, particularly for streaming data. The real-time aspect is dependent on how frequent the data bases are often refreshed with data in some cases it can vary hourly, daily, weekly.

**Impact of Poor Data Quality and Processing.**

1. Poor Decision making. Inaccurate or incomplete data can lead to flawed decision-making, resulting in wasted resources and missed opportunities [55].
2. Operational inefficiencies. Poor data quality can lead to operational inefficiencies that can lead to poor decision making that is negative for organisational development. Departments that can be affected because of inaccurate data include sales, marketing, and research and development among others [55].
3. Compliance Issues. Inaccurate data can make it difficult for organizations to comply with regulatory requirements and standards [55].

## **Use cases of KPI automation**

KPI automation by leveraging python and various data sources has transformative applications across various organisational departments enabling data driven decisions tailored to specific organisation functions [56]. Common use cases include automating data collection, generating reports, and providing insights into performance, including software development, IT operations, business intelligence [57]. Here is a breakdown of business intelligence application as it will be the use case we are focusing on.

In **sales**, automated PowerPoint reports can visualize the latest metrics which can include, win rates, and revenue among others. These reports can automatically highlight underperforming accounts with conditional formatting, maps sales illustrating heat maps of highest selling regions and also for better visualization of sales data.

**Marketing** teams benefit from automated PowerPoint reports by tracking real-time ROI (return on interest), customer acquisition costs, and different conversion rates among others. Marketing departments can leverage such tools t generate reports that can use for internal and external management.

**Finance** departments leverage these tools to generate monthly performance decks featuring interactive P&L statements (Profit and loss), cash flow analyses collected form various data sources.

**Operations** can automate production efficiency which usually show metrics such as success rates, conversion rates, and quantity demanded / dispatched among others.

Moreso, **research and development** can also benefit from the tool to automated task regarding product performance or KPIs relating to different R&D projects that can improve efficiency by reducing time spent in manually computing them.

## **Demand for KPI Automated reports**

The demand for automated KPI reporting is experiencing significant growth, with industry analysts projecting near-universal adoption across mid to large enterprises by 2030. The current estimates suggest that over 65% of global organizations with 200+ employees already use some form of automated reporting, a figure expected to be above 90% by 2027. This trajectory is driven by several factors which include [58];

Firstly, the pandemic accelerated digital transformation, with 71% of companies prioritizing analytics automation to support remote decision making [59]. Secondly, increase in data and data sources has made manual report generations very redundant and unsustainable. KPI tracking helps businesses develop data minded culture which often times helps outperform their competitors this allows different managers and stake holders make strategic decisions to influence/ drive their businesses in the right direction [60].

As a result of data growing rapidly each year at a rate of 60% - 70%, the compound annual growth rate for storage capacity is forecast to be 19.2% from 2020 to 2025. This means that the total amount of data created, captured, copied, and consumed globally is increasing by 19.2% each year [61]. As a result, CRM (customer relationship management) tools have limitations in terms of processing speed, memory, and data storage making manual KPI report generation very difficult and most likely redundant. While excel for example can handle moderately sized data sized data sets, its performance degrades when dealing with significantly large data sets [62].

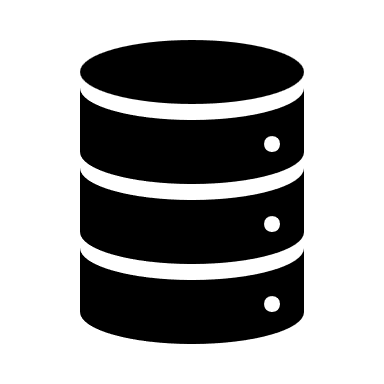
# **Methodology**

Using a secondary data source, we shall develop a dash-board that allows a user to apply filters relevant to their use case and generate a KPI Power Point report that can be used for stake holder management.

**Secondary data** which will mostly consist of open-source data usually in excel work books will substitute company data base as a result of access limitations. The data will contain multiple variables and data points which will then be aggregated using python and presenting into a report. Use of statistical methods such as data aggregation functions, graphical representations will be done to illustrate the trends that lie with in the data for different periods of time.

## **System Architecture Overview**

**Input** **Processing**  **Output**



User interface to generate Report

Using python to aggregate data



[**A**] – Input which is usually consisting of excel workbook/data base which will be used in the illustration.

[**B**] – represents the output of the entire process from data acquisition to data aggregation and finally to data visualization.

[**C**] – Is an input PowerPoint file containing place holders that allows us to dynamically change the positions of where the KPIs will be placed. It is important to note that the input power point file is arbitrary and can vary depending on use case hence can be customized according to preference.

**Procedure.**

**Data collection**. Collect data from applicable sources such as (Company database, excel workbooks, API).

**KPI calculation and computation.** Use statistical methods to aggregate large sets of data for metric generation [63].

**User interface.** Using libraries like plotly dash, python pptx, pandas matplotlib to generate graphs necessary for visualizing trends [64].

**Power Generation.** Develop a user interface using Plotly dash to enable users to apply appropriate filters and download the KPIs needed [64].

## **KPI Definition and Formation**

For this use case, open-source sales e-commerce data stored in an excel work book will be used demonstrate automated KPI report generation. The data is mainly comprises of the following variables [65];

* Age
* New\_user
* Total\_pages\_visited
* Converted

Considering the following variables, the following relevant KPIs can be extracted assess possible factors affecting sales.

**Age demographic.** This will help us understand the different age categories consumers fall under and their buying behaviours. For online web analysis and marketing, the KPI provides deeper insights on the websites target audience [66].

**New\_users.** We shall get a count of the number of new users visiting the website and understand the different conversion rates. The mathematical operation to compute this will be represented by the following equation as shown below [67].

**Converted.** This will give us an understanding on the total number website visits that resulted into sales and it is illustrated by the following formula as shown below.

**Conversion rates.** The conversion rates will be calculated by the following mathematical formula given below [68].

**Analytics overview**. The analytics overview comprises of graphical representation showing trends regarding the sales which comprise of;

1. Total sites visited by age groups
2. Conversion rate by age group

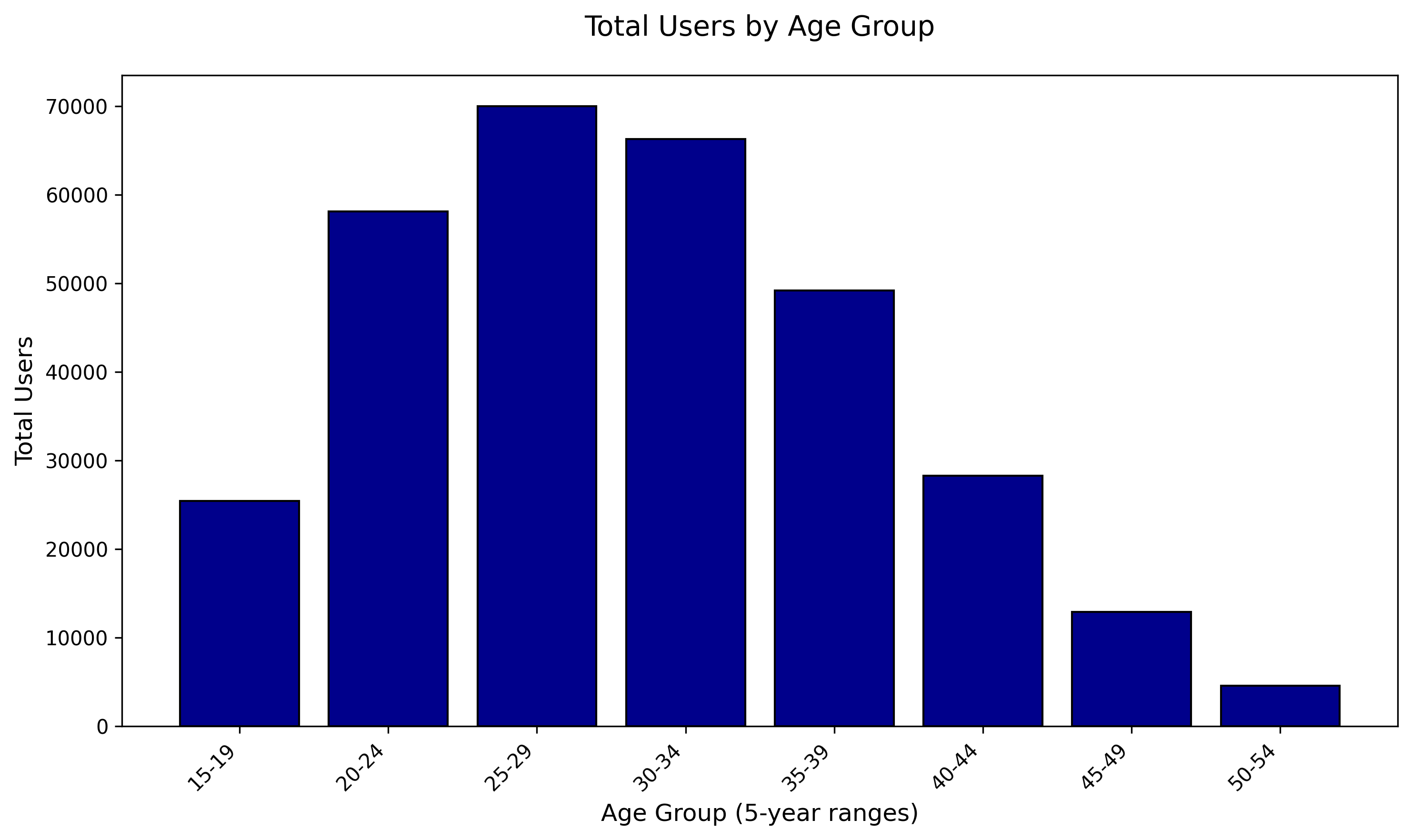


Figure 1 Total webusers by age category

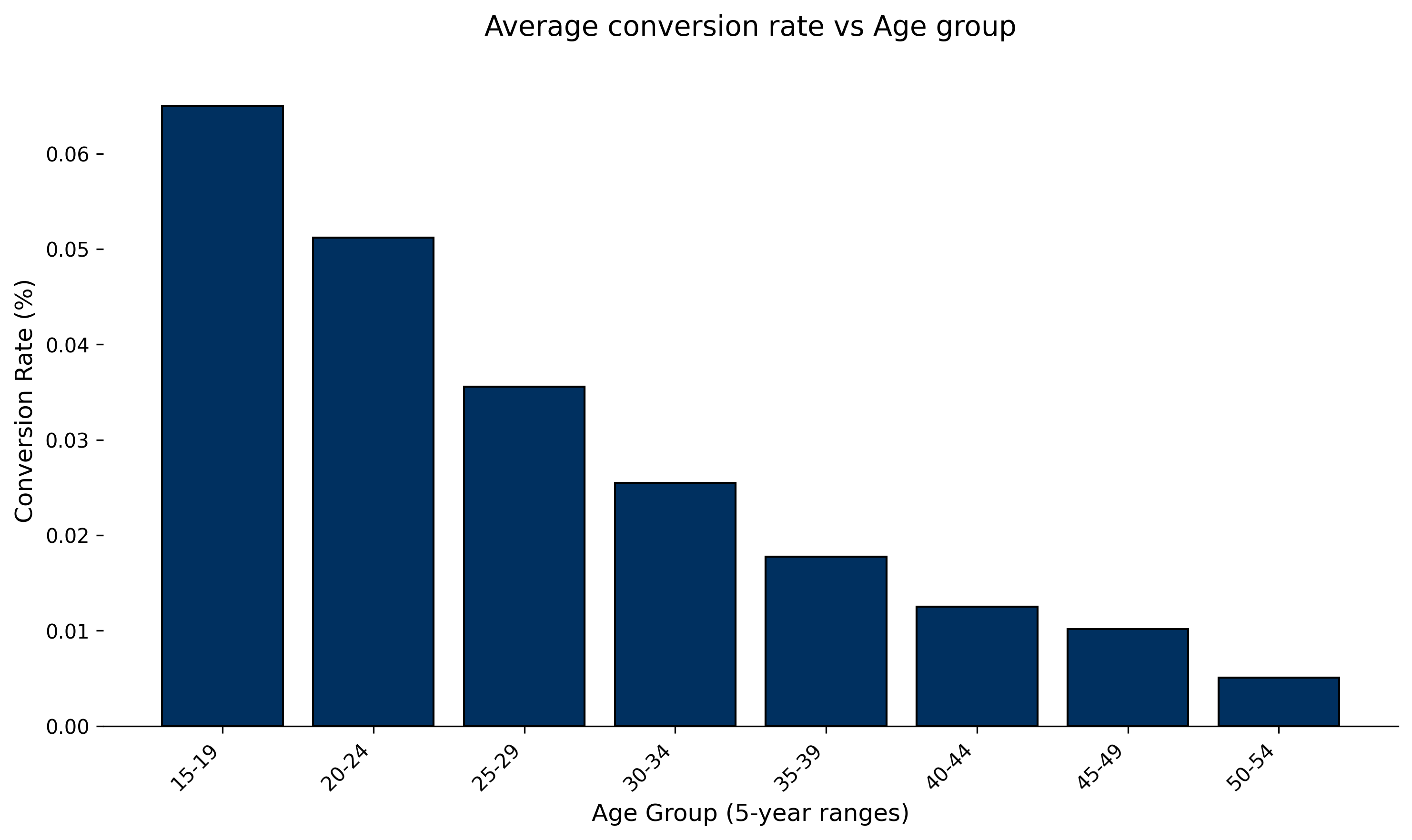


Figure 2 Showing conversion rate by age category

## **User Interface**

The user interface/ front end of the dash board will display the KPIs and graph visualizations that can be exported for further use reducing time spent in customizing the PowerPoint presentations.

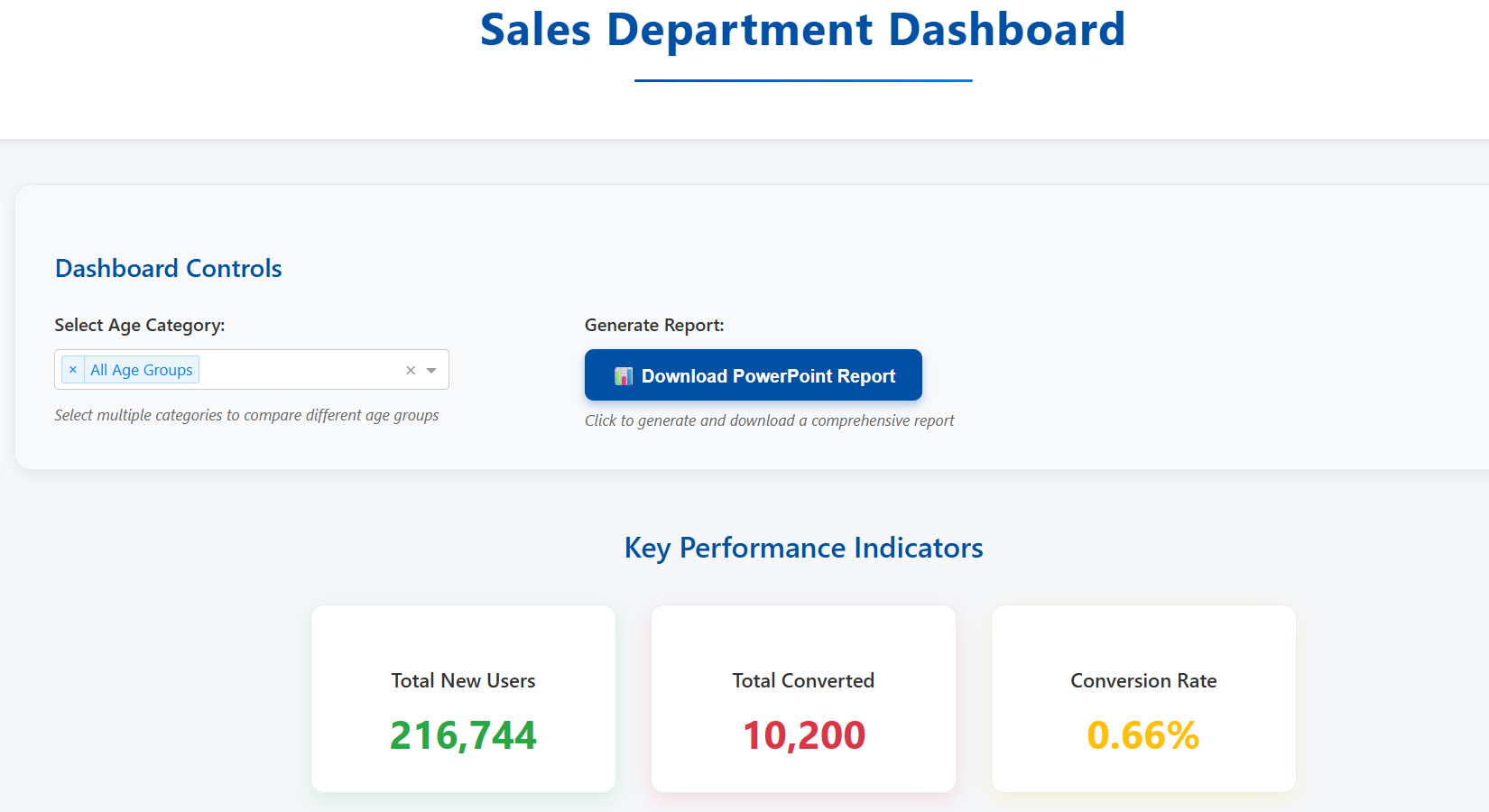


Figure 3 Showing an example of the dashboard layout

## 

The image shows the layout of the dashboard the provides a brief summary of the sales performance data. Users can export these KPIs together with graphical representations by simply pressing the download Power Point report. Depending on what age categories users are interested in analysing they can get quick summary of the KPIs by simply making use of the “Select Age Category” filter.

## **Processing data with python**

For data processing, it is important to import libraries that provide assistance when handling data. Pandas specifically is used in this research paper to read and manipulate data for further analysis to be done. Additionally, libraries such as subprocess, matplotlib among others are used to aggregate data and also produce vectorized images/ plots to have enhanced visuals when the PowerPoints are exported. In order to test out the illustration for first time users libraries can be imported using “pip install library name”. This command prompt can be initiated in the terminal and the respective libraries will be installed [69].

import subprocess

import matplotlib.ticker as ticker

import io

import os

import matplotlib.pyplot as plt

import pandas as pd

from datetime import datetime

from pptx import Presentation

from pptx.util import Inches, Pt

from pptx.dml.color import RGBColor

from pptx.enum.text import PP\_ALIGN

from pptx.enum.shapes import MSO\_SHAPE

import dash

from pptx.util import Inches

from dash import dcc, html, no\_update

from dash.dependencies import Input, Output, State

The function below checks the input file to ensure that there is data before processing and analysis is done. This process allows us to detect errors by returning an empty list of there is no existent data [70].

Moreso, the data used for the KPI generation was collected from an open-source data source. The raw data comprised mainly of:

* Structured data: This mainly consists of CSV files and SQL tables among others containing metrics such as age, new\_user, total\_pages\_visited, and converted [71].

To successfully automated generation of KPI Power Point reports, intermediate steps were considered which comprise of the following;

1. **Data reading**. The data was read and checked to ensure that there were no missing values and so compute the KPI while avoiding any computational errors. This done by using the “pd.read\_csv” pandas attribute as shown below [72].

#data is read from here can also be a to a data base depending on the use case we shall use an excel work book

df1 = pd.read\_csv('online\_sales.csv', delimiter = ',')

1. **Data grouping.** The data contains a range of various ages and corresponding variables. To get a better understanding age categories were created for easier comparison, analysis, and interpretation of data especially for large data sets. The following functions shows how different age ranges that are subdivided and categorised [73].

def get\_age\_categories(df1):

    if df1.empty:

        return []

    # Get min and max ages, rounded down to nearest 5

    min\_age = (min(df1['age']) // 5) \* 5

    max\_age = (max(df1['age']) // 5) \* 5

    # Create age ranges in steps of 5

    age\_bins = range(min\_age, max\_age + 6, 5)

    # Build the age category labels

    age\_labels = []

    for i in age\_bins[:-1]:

        category = str(i) + "-" + str(i + 4)

        age\_labels.append(category)

    # Convert to the expected format

    result = []

    for category in age\_labels:

        result.append({'label': category, 'value': category})

    return result

1. **Ppt layout orientation**. For the presentation to be automatically generated, essential functions such as the one below are used to custom fill shapes with colors based on the user’s preference. The “fill” function is used to distinguish from line, the boarder around a shape, which have a different colour [74].

# Function to define ppt layout and specifications (where KPIs are placed)

def set\_custom\_fill\_and\_outline(shape, is\_large\_rectangle = False):

    if is\_large\_rectangle:

        fill\_color = RGBColor(244,244,244)

    else:

        fill\_color = RGBColor(222,222,223)

    shape.fill.solid()

    shape.fill.fore\_color.rgb = fill\_color

    shape.line.color.rgb = fill\_color

1. **Data aggregation**. Data is aggregated using different mathematical functions to get a summary of what the data entails. The mathematical functions used in aggregation are described “KPI definition and formation” the aggregates done are illustrated by the following function [75]. For this research paper KPIs such as total new users, total converted, total pages visited and total conversion rates among others are calculated.

#Function to calculate KPIs

def calculate\_kpis(df1):

    if df1.empty:

        return 0,0,0

    total\_new\_users = df1['new\_user'].sum()

    total\_converted = df1['converted'].sum()

    total\_pages\_visited = df1['total\_pages\_visited'].sum()

    total\_conversion = df1['converted'].sum()

    conversion\_rate = round((total\_conversion/total\_pages\_visited)\*100,2) if total\_pages\_visited > 0 else 0

    return total\_new\_users, total\_converted, conversion\_rate

1. **Data visualization**. Data is visualized using graphs to identify trends and patterns that offer better perception on the data being analysed. Two graphs were plotted one visualising the “Total sites by age group” and “Conversion rate by age group” as represented in figure 1 and 2 respectively . The two graphs plotted are plotted using libraries such as matplotlib [76].

**Vectorized** graphs are plotted because they are non pixelated hence providing better visuals when presenting the KPIs. A summary of the aggregated data is also added to the Power Point report using the code represented below.

The function below illustrates how a vectorized graph showing total sites by age group.

def generate\_total\_sites\_chart(df1, title\_suffix=""):

    try:

        # Check if we have any data to work with

        if df1.empty:

            print("DataFrame is empty, cannot generate chart")

            return None

        # Calculate age ranges

        min\_age = (min(df1['age']) // 5) \* 5

        max\_age = (max(df1['age']) // 5) \* 5

        age\_bins = range(min\_age, max\_age + 6, 5)

        age\_labels = []

        for i in age\_bins[:-1]:

            label = str(i) + "-" + str(i + 4)

            age\_labels.append(label)

        # Work with a copy to avoid modifying the original dataframe

        df1\_copy = df1.copy()

        # Create age groups using pandas cut function

        df1\_copy['AgeGroup'] = pd.cut(df1\_copy['age'], bins=age\_bins, labels=age\_labels, right=False)

        # Group by age and sum up total pages visited

        age\_group\_stats = df1\_copy.groupby('AgeGroup')['total\_pages\_visited'].sum().reset\_index()

        # Set up the plot

        plt.figure(figsize=(10, 6))

        bars = plt.bar(age\_group\_stats['AgeGroup'], age\_group\_stats['total\_pages\_visited'],

                       color='skyblue', edgecolor='black')

        # Add numbers on top of each bar

        for bar in bars:

            height = bar.get\_height()

            x\_pos = bar.get\_x() + bar.get\_width() / 2.0

            plt.text(x\_pos, height, str(int(height)), ha='center', va='bottom')

        # Set up the title

        if title\_suffix:

            title = "Total pages visited vs Age group " + title\_suffix

        else:

            title = "Total pages visited vs Age group"

        title = title.strip()

        # Customize the chart appearance

        plt.title(title, fontsize=14, pad=20)

        plt.xlabel('Age Group (5-year ranges)', fontsize=12)

        plt.ylabel('Total Pages Visited', fontsize=12)

        plt.grid(axis='y', linestyle='', alpha=0.7)

        plt.xticks(rotation=45, ha='right')

        plt.tight\_layout()

        # Remove chart borders/spines

        ax = plt.gca()

        ax.spines['top'].set\_visible(False)

        ax.spines['right'].set\_visible(False)

        ax.spines['left'].set\_visible(False)

        ax.spines['bottom'].set\_visible(True)  # Keep x-axis visible

        # Set up file paths

        svg\_path = 'temp\_chart.svg'

        emf\_path = 'output\_chart.emf'

        png\_path = 'output\_chart.png'

        # Save as SVG first

        plt.savefig(svg\_path, format='svg', bbox\_inches='tight', dpi=600, transparent=True)

        # Try to convert SVG to EMF format

        result\_path = convert\_svg\_to\_emf(svg\_path, emf\_path)

        # Check if EMF conversion worked

        if result\_path is None or not os.path.exists(result\_path):

            # EMF didn't work, use PNG instead

            print("Using PNG fallback")

            plt.savefig(png\_path, format='png', bbox\_inches='tight', dpi=300, transparent=True)

            plt.close()

            # Clean up the temporary SVG file

            if os.path.exists(svg\_path):

                os.remove(svg\_path)

            # Check if PNG was created successfully

            if os.path.exists(png\_path):

                print("Chart saved as PNG: " + png\_path)

                return png\_path

            else:

                print("Failed to save PNG chart")

                return None

        # EMF worked, close the plot and clean up

        plt.close()

        print("Chart saved as EMF: ", result\_path)

        return result\_path

    except Exception as e:

        print("Error generating chart: " + str(e))

        plt.close()

        return None

A similar function to generate the conversion rate by Age group is written and it is represented by the code block below.

def generate\_conversion\_chart(df1, title\_suffix=""):

    try:

        # Check if we have data to work with

        if df1.empty:

            print("DataFrame is empty, cannot generate chart")

            return None

        # Calculate age ranges - round down to nearest 5

        min\_age = (min(df1['age']) // 5) \* 5

        max\_age = (max(df1['age']) // 5) \* 5

        age\_bins = range(min\_age, max\_age + 6, 5)

        age\_labels = []

        for i in age\_bins[:-1]:

            label = str(i) + "-" + str(i + 4)

            age\_labels.append(label)

        # Work with a copy to avoid modifying original dataframe

        df1\_copy = df1.copy()

        df1\_copy['AgeGroup'] = pd.cut(df1\_copy['age'], bins=age\_bins, labels=age\_labels, right=False)

        # Calculate average conversion rate by age group

        age\_group\_stats = df1\_copy.groupby('AgeGroup')['converted'].mean().reset\_index()

        # Create the plot

        plt.figure(figsize=(10, 6))

        bars = plt.bar(age\_group\_stats['AgeGroup'], age\_group\_stats['converted'],

                      color='#003060', edgecolor='black')

        # Set up the title

        if title\_suffix:

            title = "Average conversion rate vs Age group " + title\_suffix

        else:

            title = "Average conversion rate vs Age group"

        title = title.strip()

        # Customize the chart

        plt.title(title, fontsize=14, pad=20)

        plt.xlabel('Age Group (5-year ranges)', fontsize=12)

        plt.ylabel('Conversion Rate (%)', fontsize=12)

        plt.grid(axis='y', linestyle='', alpha=0.3)

        plt.xticks(rotation=45, ha='right')

        plt.tight\_layout()

        # Remove chart borders

        ax = plt.gca()

        ax.spines['top'].set\_visible(False)

        ax.spines['right'].set\_visible(False)

        ax.spines['left'].set\_visible(False)

        ax.spines['bottom'].set\_visible(True)

        # Set up file paths

        svg\_path = 'temp\_conversion\_chart.svg'

        emf\_path = 'conversion\_chart.emf'

        png\_path = 'conversion\_chart.png'

        # Save as SVG first

        plt.savefig(svg\_path, format='svg', bbox\_inches='tight', dpi=600, transparent=True)

        # Try to convert to EMF

        result\_path = convert\_svg\_to\_emf(svg\_path, emf\_path)

        # Check if EMF conversion worked

        if result\_path is None or not os.path.exists(result\_path):

            # Use PNG as fallback

            print("Using PNG fallback for conversion chart...")

            plt.savefig(png\_path, format='png', bbox\_inches='tight', dpi=300, transparent=True)

            plt.close()

            # Clean up temporary SVG file

            if os.path.exists(svg\_path):

                os.remove(svg\_path)

            # Check if PNG was created successfully

            if os.path.exists(png\_path):

                print("Conversion chart saved as PNG: " + png\_path)

                return png\_path

            else:

                print("Failed to save PNG conversion chart")

                return None

        # EMF worked, clean up and return

        plt.close()

        print("Conversion chart saved as EMF: " + result\_path)

        return result\_path

    except Exception as e:

        print("Error generating conversion chart: " + str(e))

        plt.close()

        return None

Additionally, three functions are used to create the layout of the presentation. And these functions include the following.

**Add KPI function**. This is specifically used to add KPIs to the power point presentation when the exporting the PowerPoint file. The add KPI function can be dynamically adjusted depending on user /developer preference and for this particular research paper the functions’ main objective is adjust orientation of text as well colour of the texts being exported in the power point file [77].

#Function to add KPI metrics to slides

def add\_kpi(slide, left, top, value, label):

    text\_box = slide.shapes.add\_textbox(left, top, Inches(2), Inches(1))

    text\_frame = text\_box.text\_frame

    text\_frame.clear()

    p = text\_frame.paragraphs[0]

    run = p.add\_run()

    run.text = value

    run.font.size = Pt(16)

    run.font.bold = True

    run.font.color.rgb = RGBColor(0, 51, 102)

    p = text\_frame.add\_paragraph()

    run = p.add\_run()

    run.text = label

    run.font.size = Pt(11)

    run.font.color.rgb = RGBColor(0, 51, 102)

**Add heading text function**. The heading text function is used to generate headings for respective ppt slides as well change the orientation of the headings to best suit the applicability PowerPoint Presentation [78].

#Function to add the Key statistics heading

def add\_heading\_text(slide, left, top, text):

    textbox = slide.shapes.add\_textbox(left, top, Inches(3), Inches(0.5))

    text\_frame = textbox.text\_frame

    text\_frame.text = text

    paragraph = text\_frame.paragraphs[0]

    paragraph.alignment = PP\_ALIGN.CENTER

    paragraph.font.size = Pt(18)

    paragraph.font.bold = True

    textbox.left = int(left - (textbox.width / 2))

**Clone shapes function**. The clone shape’s function is used to duplicate the shapes within the PowerPoint slides. It clones the shapes of the second slide of the input ppt file to allow duplication in case further slides are needed [79].

Function to clone shapes from source slide

def clone\_shapes(source\_slide, new\_slide):

    for shape in source\_slide.shapes:

        if not shape.is\_placeholder:

            if shape.shape\_type == MSO\_SHAPE.RECTANGLE:

                new\_shape = new\_slide.shapes.add\_shape(

                    shape.auto\_shape\_type,

                    shape.left,

                    shape.top,

                    shape.width,

                    shape.height

                )

                is\_large\_rectangle = shape.width > Inches(5.5)

                set\_custom\_fill\_and\_outline(new\_shape, is\_large\_rectangle)

                if hasattr(new\_shape, 'text\_frame'):

                    new\_shape.\_element.remove(new\_shape.\_element.txBody)

            elif shape.shape\_type == MSO\_SHAPE.ROUNDED\_RECTANGLE:

                new\_shape = new\_slide.shapes.add\_shape(

                    MSO\_SHAPE.ROUNDED\_RECTANGLE,

                    shape.left,

                    shape.top,

                    shape.width,

                    shape.height

                )

                is\_large\_rectangle = shape.width > Inches(5.5)

                set\_custom\_fill\_and\_outline(new\_shape, is\_large\_rectangle)

                if hasattr(new\_shape, 'text\_frame'):

                    new\_shape.\_element.remove(new\_shape.\_element.txBody)

            elif shape.shape\_type == 13:  # Picture shape

                image\_stream = shape.image.blob

                image\_file = io.BytesIO(image\_stream)

                new\_slide.shapes.add\_picture(

                    image\_file,

                    shape.left,

                    shape.top,

                    shape.width,

                    shape.height

                )

Lastly, a function to add the KPIS to the Power Point is also used to such that we get a summary of the previously calculated data as shown below.

def add\_charts\_to\_presentation(prs, df\_filtered, slide\_index=1):

    try:

        # Check if the slide index is valid

        if len(prs.slides) <= slide\_index:

            error\_msg = "Slide index " + str(slide\_index) + " out of range (presentation has " + str(len(prs.slides)) + " slides)"

            raise ValueError(error\_msg)

        # Calculate the KPI values

        print("Calculating KPIs...")

        total\_new\_users, total\_converted, conversion\_rate = calculate\_kpis(df\_filtered)

        print("KPIs calculated - New Users:", total\_new\_users, "Converted:", total\_converted, "Rate:", conversion\_rate, "%")

        # Get the slide we want to work with

        slide = prs.slides[slide\_index]

        print("Working on slide", slide\_index)

        # Update KPI placeholders

        print("Updating KPI placeholder")

        kpis\_updated = False

        for shape in slide.shapes:

            if shape.has\_text\_frame:

                text = shape.text\_frame.text.strip()

                if text == 'A':

                    shape.text\_frame.text = "Total new users: " + str(total\_new\_users)

                    kpis\_updated = True

                    print("Updated placeholder A with:", total\_new\_users)

                elif text == 'B':

                    shape.text\_frame.text = "Total converted:" + str(total\_converted)

                    kpis\_updated = True

                    print("Updated placeholder B with:", total\_converted)

                elif text == 'C':

                    shape.text\_frame.text = "Conversion rate:" + str(conversion\_rate) + "%"

                    kpis\_updated = True

                    print("Updated placeholder C with:", conversion\_rate, "%")

        # Check if we found any KPI placeholders

        if not kpis\_updated:

            print("No KPI placeholders (A, B, C) were found on the input ppt slide - check the input the slide")

        # Generate both charts

        print("Generating charts.")

        chart\_path = generate\_total\_sites\_chart(df\_filtered, title\_suffix="")

        chart\_path1 = generate\_conversion\_chart(df\_filtered, title\_suffix="")

        # Add the first chart if it was created successfully

        if chart\_path and os.path.exists(chart\_path):

            print("Total sites chart generated at:", chart\_path)

            try:

                # Set position for first chart (left side)

                left = Inches(0.5)

                top = Inches(3.0)

                width = Inches(6)

                height = Inches(4)

                # Add the picture to the slide

                pic = slide.shapes.add\_picture(chart\_path, left, top, width, height)

                print("Total sites chart added to slide successfully")

                # Look for chart title placeholder and update it

                title\_updated = False

                for shape in slide.shapes:

                    if shape.has\_text\_frame and shape.text\_frame.text.strip() == 'D':

                        shape.text\_frame.text = ""

                        title\_updated = True

                        break

                if not title\_updated:

                    print("Chart title placeholder 'D' not found")

            except Exception as e:

                print("Error adding total sites chart to slide:", e)

                return None

        else:

            print("Error: Total sites chart not generated or file not found")

            return None

        # Add the second chart if it was created successfully

        if chart\_path1 and os.path.exists(chart\_path1):

            print("Conversion chart generated at:", chart\_path1)

            try:

                # Set position for second chart (right side)

                left = Inches(7)

                top = Inches(3.0)

                width = Inches(6)

                height = Inches(4)

                print("Adding conversion chart to slide at position: " + str(left.inches) + "in, " + str(top.inches) + "in")

                # Add the picture to the slide

                pic = slide.shapes.add\_picture(chart\_path1, left, top, width, height)

                print("Conversion chart added to slide successfully")

                # Look for chart title placeholder and update it

                title\_updated = False

                for shape in slide.shapes:

                    if shape.has\_text\_frame and shape.text\_frame.text.strip() == 'E':

                        shape.text\_frame.text = ""

                        title\_updated = True

                        break

                if not title\_updated:

                    print("Chart title placeholder 'E' not found")

            except Exception as e:

                print("Error adding conversion chart to slide:", e)

                return None

        else:

            print("Error: Conversion chart not generated")

            return None

        print("Charts added successfully")

        # Clean up all temporary files

        temp\_files = [

            'temp\_conversion\_chart.svg', 'conversion\_chart.emf', 'conversion\_chart.png',

            'temp\_total\_sites\_chart.svg', 'total\_sites\_chart.emf', 'total\_sites\_chart.png'

        ]

        for temp\_file in temp\_files:

            if os.path.exists(temp\_file):

                try:

                    os.remove(temp\_file)

                    print("Removed temporary file:", temp\_file)

                except Exception as e:

                    print("Warning: Could not remove", temp\_file, ":", e)

        return prs

    except Exception as e:

        print("Critical error in add\_charts\_to\_presentation:", e)

        import traceback

        traceback.print\_exc()

        return None

**5. Dash Layout**. The dash layout represents the front end/ user interface that will allow different users to apply filters allowing them to export ad hoc KPI reports depending on different use cases. In this paper sales data is analysed to get a general summary of the performance of an ecommerce platform.

Reasons why these KPIs are selected is based on the data provided in order for sales performance on an ecommerce platform to be assessed and derive conclusive results for rational decision making.

1. **Total new users**. This is a KPI will allow different stake holders know the total number of users visiting their ecommerce platforms.
2. **Total converted**. This KPI shows the total number site visits that were converted in sales on the ecommerce platform.
3. **Conversion rate**. This KPI provides a general understanding of the rate at which site visits are being converted into sales.

For the dash layout / front end to be implemented successfully the following needs to considered when writing the code.

**Initialization** of the app is done because it this serves as the foundation for building interactive web-based dash boards as show below [80].

app = dash.Dash(\_\_name\_\_)

**Role in the layout.** The app.layout object is used and it is a very crucial structure because it holds the UI (User Interface) structure and can be dynamically adjusted and changed according to requirements. It is one of the most important features because it allows the designing of the dash board user interface by allowing implementation of features such as headings, graphs, dropdowns, and download buttons among others. For this use case the dash app.layout is represented by the following block of code below [81].

# Create the Dash layout with professional styling

app.layout = html.Div([

    # Header Section

    html.Div([

        html.H1("Sales Department Dashboard",

                style={

                    'color': '#0051a6',

                    'textAlign': 'center',

                    'marginBottom': '0px',

                    'fontSize': '2.5rem',

                    'fontWeight': '700',

                    'letterSpacing': '1px'

                }),

        html.Hr(style={

            'border': 'none',

            'height': '3px',

            'background': 'linear-gradient(90deg, #0051a6, #007bff)',

            'margin': '20px auto',

            'width': '300px',

            'borderRadius': '2px'

        })

    ], style={

        'backgroundColor': 'white',

        'padding': '40px 20px 30px 20px',

        'boxShadow': '0 2px 10px rgba(0,0,0,0.1)',

        'marginBottom': '40px'

    }),

    # Main Content Container

    html.Div([

        # Control Panel Section

        html.Div([

            html.H3("Dashboard Controls",

                   style={

                       'color': '#0051a6',

                       'marginBottom': '25px',

                       'fontSize': '1.4rem',

                       'fontWeight': '600'

                   }),

            html.Div([

                # Age Category Filter Section

                html.Div([

                    html.Label("Select Age Category:",

                              style={

                                  'fontWeight': '600',

                                  'marginBottom': '12px',

                                  'color': '#333',

                                  'fontSize': '1rem',

                                  'display': 'block'

                              }),

                    dcc.Dropdown(

                        id='age-category-dropdown',

                        options=[{'label': 'All Age Groups', 'value': 'all'}] + get\_age\_categories(df1),

                        value='all',

                        multi=True,

                        placeholder="Choose age categories...",

                        style={

                            'width': '350px',

                            'marginBottom': '10px',

                            'fontSize': '0.95rem'

                        }

                    ),

                    html.Small("Select multiple categories to compare different age groups",

                             style={

                                 'color': '#666',

                                 'fontStyle': 'italic',

                                 'fontSize': '0.85rem'

                             })

                ], style={

                    'display': 'inline-block',

                    'marginRight': '80px',

                    'verticalAlign': 'top',

                    'minWidth': '350px'

                }),

                # Download Section

                html.Div([

                    html.Label("Generate Report:",

                              style={

                                  'fontWeight': '600',

                                  'marginBottom': '12px',

                                  'color': '#333',

                                  'fontSize': '1rem',

                                  'display': 'block'

                              }),

                    html.Button("📊 Download PowerPoint Report",

                               id="download-btn",

                               n\_clicks=0,

                               style={

                                   'backgroundColor': '#0051a6',

                                   'color': 'white',

                                   'border': 'none',

                                   'padding': '12px 24px',

                                   'fontSize': '1rem',

                                   'fontWeight': '600',

                                   'borderRadius': '8px',

                                   'cursor': 'pointer',

                                   'transition': 'all 0.3s ease',

                                   'boxShadow': '0 4px 8px rgba(0,81,166,0.3)',

                                   'minWidth': '250px'

                               }),

                    dcc.Download(id="download-ppt"),

                    html.Small("Click to generate and download a comprehensive report",

                             style={

                                 'color': '#666',

                                 'fontStyle': 'italic',

                                 'fontSize': '0.85rem',

                                 'display': 'block',

                                 'marginTop': '8px'

                             })

                ], style={

                    'display': 'inline-block',

                    'verticalAlign': 'top'

                }),

            ], style={

                'display': 'flex',

                'flexWrap': 'wrap',

                'alignItems': 'flex-start',

                'gap': '40px'

            }),

        ], style={

            'backgroundColor': '#f8f9fa',

            'padding': '35px',

            'borderRadius': '15px',

            'marginBottom': '50px',

            'boxShadow': '0 4px 12px rgba(0,0,0,0.08)',

            'border': '1px solid #e9ecef'

        }),

        # KPI Cards Section

        html.Div([

            html.H3("Key Performance Indicators",

                   style={

                       'color': '#0051a6',

                       'textAlign': 'center',

                       'marginBottom': '35px',

                       'fontSize': '1.6rem',

                       'fontWeight': '600'

                   }),

            html.Div([

                # New Users Card

                html.Div([

                    html.Div([

                        html.I(className="fas fa-user-plus", style={

                            'fontSize': '2rem',

                            'color': '#28a745',

                            'marginBottom': '15px'

                        }),

                        html.H4("Total New Users", style={

                            'color': '#333',

                            'marginBottom': '12px',

                            'fontSize': '1.1rem',

                            'fontWeight': '600'

                        }),

                        html.H2(id="kpi-new-users", style={

                            'color': '#28a745',

                            'margin': '0',

                            'fontSize': '2.2rem',

                            'fontWeight': '700'

                        })

                    ], style={'textAlign': 'center'})

                ], style={

                    'backgroundColor': 'white',

                    'padding': '30px 25px',

                    'borderRadius': '12px',

                    'boxShadow': '0 6px 20px rgba(40,167,69,0.15)',

                    'border': '1px solid #e9ecef',

                    'minWidth': '220px',

                    'transition': 'transform 0.3s ease, box-shadow 0.3s ease'

                }),

                # Converted Users Card

                html.Div([

                    html.Div([

                        html.I(className="fas fa-check-circle", style={

                            'fontSize': '2rem',

                            'color': '#dc3545',

                            'marginBottom': '15px'

                        }),

                        html.H4("Total Converted", style={

                            'color': '#333',

                            'marginBottom': '12px',

                            'fontSize': '1.1rem',

                            'fontWeight': '600'

                        }),

                        html.H2(id="kpi-converted", style={

                            'color': '#dc3545',

                            'margin': '0',

                            'fontSize': '2.2rem',

                            'fontWeight': '700'

                        })

                    ], style={'textAlign': 'center'})

                ], style={

                    'backgroundColor': 'white',

                    'padding': '30px 25px',

                    'borderRadius': '12px',

                    'boxShadow': '0 6px 20px rgba(220,53,69,0.15)',

                    'border': '1px solid #e9ecef',

                    'minWidth': '220px',

                    'transition': 'transform 0.3s ease, box-shadow 0.3s ease'

                }),

                # Conversion Rate Card

                html.Div([

                    html.Div([

                        html.I(className="fas fa-percentage", style={

                            'fontSize': '2rem',

                            'color': '#ffc107',

                            'marginBottom': '15px'

                        }),

                        html.H4("Conversion Rate", style={

                            'color': '#333',

                            'marginBottom': '12px',

                            'fontSize': '1.1rem',

                            'fontWeight': '600'

                        }),

                        html.H2(id="kpi-conversion-rate", style={

                            'color': '#ffc107',

                            'margin': '0',

                            'fontSize': '2.2rem',

                            'fontWeight': '700'

                        })

                    ], style={'textAlign': 'center'})

                ], style={

                    'backgroundColor': 'white',

                    'padding': '30px 25px',

                    'borderRadius': '12px',

                    'boxShadow': '0 6px 20px rgba(255,193,7,0.15)',

                    'border': '1px solid #e9ecef',

                    'minWidth': '220px',

                    'transition': 'transform 0.3s ease, box-shadow 0.3s ease'

                })

            ], style={

                'display': 'flex',

                'justifyContent': 'center',

                'flexWrap': 'wrap',

                'gap': '30px'

            }),

        ], style={'marginBottom': '50px'}),

        # Update the Analytics Chart Section in your app.layout

html.Div([

    html.H3("Analytics Overview",

           style={

               'color': '#0051a6',

               'textAlign': 'center',

               'marginBottom': '30px',

               'fontSize': '1.6rem',

               'fontWeight': '600'

           }),

    html.Div([

        # First Chart - Total Sites Visited

        html.Div([

            dcc.Graph(

                id="age-chart",

                config={

                    'displayModeBar': True,

                    'displaylogo': False,

                    'modeBarButtonsToRemove': ['pan2d', 'lasso2d', 'select2d']

                }

            )

        ], style={

            'backgroundColor': 'white',

            'borderRadius': '12px',

            'padding': '20px',

            'marginBottom': '30px',

            'boxShadow': '0 2px 8px rgba(0,0,0,0.1)'

        }),

        # Second Chart - Conversion Rate

        html.Div([

            dcc.Graph(

                id="conversion-chart",

                config={

                    'displayModeBar': True,

                    'displaylogo': False,

                    'modeBarButtonsToRemove': ['pan2d', 'lasso2d', 'select2d']

                }

            )

        ], style={

            'backgroundColor': 'white',

            'borderRadius': '12px',

            'padding': '20px',

            'boxShadow': '0 2px 8px rgba(0,0,0,0.1)'

        })

    ])

        ], style={

            'backgroundColor': 'white',

            'padding': '35px',

            'borderRadius': '15px',

            'boxShadow': '0 6px 20px rgba(0,0,0,0.1)',

            'border': '1px solid #e9ecef',

            'marginBottom': '40px'

        }),

        # Status Messages Section

        html.Div(id="status-message", style={

            'textAlign': 'center',

            'padding': '20px',

            'borderRadius': '8px',

            'marginTop': '20px'

        })

    ], style={

        'maxWidth': '1400px',

        'margin': '0 auto',

        'padding': '0 30px 40px 30px'

    })

], style={

    'backgroundColor': '#f5f7fa',

    'minHeight': '100vh',

    'fontFamily': '"Segoe UI", Tahoma, Geneva, Verdana, sans-serif'

})

**Enable Callbacks**. The @app.callback links the user inputs (e.g., dropdowns) to output (e.g., graphs) and allows reports to be downloaded and exported as required in standardized format [82].

The callback function works in the following ways.

The @app.callback decorator declares:

Output: Which component/ property to update (e.g., dcc.Graph.figure)

Input: Which component/ property triggers the update (e.g., dcc.Dropdown.value)

Example is as shown below.

# Updated callback with improved chart formatting

@app.callback(

    [Output('kpi-new-users', 'children'),

     Output('kpi-converted', 'children'),

     Output('kpi-conversion-rate', 'children'),

     Output('age-chart', 'figure'),

     Output('conversion-chart', 'figure')],

    [Input('age-category-dropdown', 'value')]

)

def update\_dashboard(selected\_age\_categories):

    # Filter the dataframe based on selected age categories

    if 'all' in selected\_age\_categories or not selected\_age\_categories:

        df\_filtered = df1

    else:

        # Handle multiple age category selections

        age\_ranges = []

        for category in selected\_age\_categories:

            age\_parts = category.split('-')

            min\_age = int(age\_parts[0])

            max\_age = int(age\_parts[1])

            age\_ranges.append((min\_age, max\_age))

        # Create filter condition for all selected age ranges

        conditions = []

        for min\_age, max\_age in age\_ranges:

            condition = (df1['age'] >= min\_age) & (df1['age'] <= max\_age)

            conditions.append(condition)

        # Combine conditions with OR logic

        if conditions:

            combined\_condition = conditions[0]

            for condition in conditions[1:]:

                combined\_condition = combined\_condition | condition

            df\_filtered = df1[combined\_condition]

        else:

            df\_filtered = df1

    # Calculate KPIs using the filtered data

    total\_new\_users, total\_converted, conversion\_rate = calculate\_kpis(df\_filtered)

    # Create both chart figures

    if not df\_filtered.empty:

        # Create age bins for the charts

        min\_age = (min(df\_filtered['age']) // 5) \* 5

        max\_age = (max(df\_filtered['age']) // 5) \* 5

        age\_bins = range(min\_age, max\_age + 6, 5)

        # Build age labels manually

        age\_labels = []

        for i in age\_bins[:-1]:

            label = str(i) + "-" + str(i + 4)

            age\_labels.append(label)

        # Work with a copy of the filtered dataframe

        df\_filtered\_copy = df\_filtered.copy()

        df\_filtered\_copy['AgeGroup'] = pd.cut(df\_filtered\_copy['age'], bins=age\_bins, labels=age\_labels, right=False)

        # First chart - Total Sites Visited

        age\_group\_stats = df\_filtered\_copy.groupby('AgeGroup')['total\_pages\_visited'].sum().reset\_index()

        sites\_figure = {

            'data': [{

                'x': age\_group\_stats['AgeGroup'].astype(str),

                'y': age\_group\_stats['total\_pages\_visited'],

                'type': 'bar',

                'marker': {

                    'color': '#0051a6',

                    'line': {'color': '#003d82', 'width': 1}

                },

                'hovertemplate': '<b>Age Group:</b> %{x}<br><b>Total Sites Visited:</b> %{y:,}<extra></extra>'

            }],

            'layout': {

                'title': {

                    'text': 'Total Sites Visited by Age Group',

                    'x': 0.5,

                    'font': {'size': 18, 'color': '#0051a6', 'family': 'Segoe UI'}

                },

                'xaxis': {

                    'title': {'text': 'Age Group', 'font': {'size': 14, 'color': '#333'}},

                    'tickfont': {'size': 12, 'color': '#666'},

                    'gridcolor': '#e9ecef'

                },

                'yaxis': {

                    'title': {'text': 'Total Sites Visited', 'font': {'size': 14, 'color': '#333'}},

                    'tickfont': {'size': 12, 'color': '#666'},

                    'gridcolor': '#e9ecef'

                },

                'plot\_bgcolor': 'white',

                'paper\_bgcolor': 'white',

                'font': {'family': 'Segoe UI'},

                'margin': {'l': 80, 'r': 40, 't': 80, 'b': 80},

                'hovermode': 'x'

            }

        }

        # Second chart - Conversion Rate

        conversion\_stats = df\_filtered\_copy.groupby('AgeGroup')['converted'].mean().reset\_index()

        conversion\_figure = {

            'data': [{

                'x': conversion\_stats['AgeGroup'].astype(str),

                'y': conversion\_stats['converted'] \* 100,

                'type': 'bar',

                'marker': {

                    'color': '#28a745',

                    'line': {'color': '#218838', 'width': 1}

                },

                'hovertemplate': '<b>Age Group:</b> %{x}<br><b>Conversion Rate:</b> %{y:.1f}%<extra></extra>'

            }],

            'layout': {

                'title': {

                    'text': 'Conversion Rate by Age Group',

                    'x': 0.5,

                    'font': {'size': 18, 'color': '#0051a6', 'family': 'Segoe UI'}

                },

                'xaxis': {

                    'title': {'text': 'Age Group', 'font': {'size': 14, 'color': '#333'}},

                    'tickfont': {'size': 12, 'color': '#666'},

                    'gridcolor': '#e9ecef'

                },

                'yaxis': {

                    'title': {'text': 'Conversion Rate (%)', 'font': {'size': 14, 'color': '#333'}},

                    'tickfont': {'size': 12, 'color': '#666'},

                    'gridcolor': '#e9ecef',

                    'ticksuffix': '%'

                },

                'plot\_bgcolor': 'white',

                'paper\_bgcolor': 'white',

                'font': {'family': 'Segoe UI'},

                'margin': {'l': 80, 'r': 40, 't': 80, 'b': 80},

                'hovermode': 'x'

            }

        }

    else:

        # Empty figures if no data available

        empty\_layout = {

            'title': {

                'text': 'No data available for selected filter',

                'x': 0.5,

                'font': {'size': 18, 'color': '#666', 'family': 'Segoe UI'}

            },

            'xaxis': {'title': {'text': 'Age Group', 'font': {'size': 14, 'color': '#333'}}},

            'yaxis': {'title': {'text': '', 'font': {'size': 14, 'color': '#333'}}},

            'plot\_bgcolor': 'white',

            'paper\_bgcolor': 'white',

            'font': {'family': 'Segoe UI'},

            'margin': {'l': 80, 'r': 40, 't': 80, 'b': 80}

        }

        sites\_figure = {'data': [], 'layout': empty\_layout.copy()}

        conversion\_figure = {'data': [], 'layout': empty\_layout.copy()}

        conversion\_figure['layout']['yaxis']['title']['text'] = 'Conversion Rate (%)'

    # Return the formatted KPI values and chart figures

    total\_users\_formatted = str(total\_new\_users) + ","

    total\_users\_formatted = "{:,}".format(total\_new\_users)

    total\_converted\_formatted = "{:,}".format(total\_converted)

    conversion\_rate\_formatted = str(conversion\_rate) + "%"

    return (

        total\_users\_formatted,

        total\_converted\_formatted,

        conversion\_rate\_formatted,

        sites\_figure,

        conversion\_figure

    )

# Callback for PowerPoint download

@app.callback(

    [Output("download-ppt", "data"),

     Output("status-message", "children")],

    [Input("download-btn", "n\_clicks")],

    [State('age-category-dropdown', 'value')],

    prevent\_initial\_call=True

)

def download\_ppt(n\_clicks, selected\_age\_categories):

    if n\_clicks > 0:

        try:

            # Filter the dataframe based on selected categories

            if 'all' in selected\_age\_categories or not selected\_age\_categories:

                df\_filtered = df1

            else:

                # Handle multiple age category selections

                age\_ranges = []

                for category in selected\_age\_categories:

                    age\_parts = category.split('-')

                    min\_age = int(age\_parts[0])

                    max\_age = int(age\_parts[1])

                    age\_ranges.append((min\_age, max\_age))

                # Create filter condition for all selected age ranges

                conditions = []

                for min\_age, max\_age in age\_ranges:

                    condition = (df1['age'] >= min\_age) & (df1['age'] <= max\_age)

                    conditions.append(condition)

                # Combine conditions with OR logic

                if conditions:

                    combined\_condition = conditions[0]

                    for condition in conditions[1:]:

                        combined\_condition = combined\_condition | condition

                    df\_filtered = df1[combined\_condition]

                else:

                    df\_filtered = df1

            # Create presentation using template

            ppt\_filename = create\_presentation(df\_filtered, template\_path="Sales\_presentation1.pptx")

            # Check if presentation was created successfully

            if ppt\_filename and os.path.exists(ppt\_filename):

                # Read the file and prepare for download

                with open(ppt\_filename, 'rb') as f:

                    ppt\_data = f.read()

                # Return success response

                success\_message = html.Div("✅ Report downloaded successfully!",

                       style={

                           'color': '#28a745',

                           'fontWeight': 'bold',

                           'backgroundColor': '#d4edda',

                           'border': '1px solid #c3e6cb',

                           'padding': '12px 20px',

                           'borderRadius': '8px',

                           'display': 'inline-block'

                       })

                return (

                    dcc.send\_bytes(ppt\_data, filename=ppt\_filename),

                    success\_message

                )

            else:

                # Return error if presentation creation failed

                error\_message = html.Div("Error generating report. Please check template file and debug output.",

                       style={

                           'color': '#dc3545',

                           'fontWeight': 'bold',

                           'backgroundColor': '#f8d7da',

                           'border': '1px solid #f5c6cb',

                           'padding': '12px 20px',

                           'borderRadius': '8px',

                           'display': 'inline-block'

                       })

                return (no\_update, error\_message)

        except Exception as e:

            # Handle any exceptions that occur

            import traceback

            error\_details = traceback.format\_exc()

            print("Download error: " + error\_details)

            error\_message = html.Div("Error: " + str(e),

                   style={

                       'color': '#dc3545',

                       'fontWeight': 'bold',

                       'backgroundColor': '#f8d7da',

                       'border': '1px solid #f5c6cb',

                       'padding': '12px 20px',

                       'borderRadius': '8px',

                       'display': 'inline-block'

                   })

            return (no\_update, error\_message)

    # Return nothing if button hasn't been clicked

    return no\_update, ""

**Run the Server.** App.run\_server() starts the local web server to render the dashboard. It is important to know that the local host cannot be used publicly because it is registered to local port that can only accessed by one device [83].

Example:

if \_\_name\_\_ == '\_\_main\_\_':

    app.run\_server(debug=True, port=8060)

By leveraging the inbuilt python libraries as listed above, the automation of KPIS can be accomplished hence improving speed, accuracy and efficiency. In this research paper, a clear illustration using ecommerce sales data is used to compute KPIS and accurately remove redundancy by alleviating manual steps and procedures that would have been done to generate KPIS by just one click of a button. For this specific use, it takes 10 seconds to automatically generate the KPI Power Point report hence improving efficiency. In order to test out the example described in this research paper, further instructions are available in the open git repository for public use.

# **Summary**

The volume of data in the world is growing at an unprecedented rate driven by advancements artificial intelligence, Internet of things (IoT), and digital transformation across industries resulting into increase in redundant work / tasks. The technological approach of KPI automation aims to streamline the processes, enhance efficiency, and reduce human error by leveraging python.

This paper aims at offering a guided procedure on how python coupled with its different libraries can be used to set up a dash board that allows users to export data based on the filters they applied.

Firstly, libraries are imported into python such as pandas, plotly, matplotlib, numpy, and pptx among others. These libraries all have a specific use that facilitates the development of the KPI dash app as shown. Moreso, different functions are used to aggregate the data and get an understanding / summary of the data as a whole. The aggregate functions used in this research paper include the total number of new users which is calculated by summing the number of new users, the total number of converted site visits which is gotten by summing the total converted sites and the conversion rates which calculated by expressing the converted sites as a ratio of the total number of new users.

Secondly, data is then expressed inform of graphs to identify different conversion rates for the different age groups. Vectorized images are created / plotted to alleviate pixels that are no ideal for power point visuals.

Furthermore, a dash layout is created allow the user apply filters and download KPI data based on the filters applied. For this research paper, the filters applicable are the different age categories that will allow different stakes holders receive real time analytics on the different age categories once applied.

The data being used in this use case is e-commerce sales data to automate the generation of crucial KPIS the sales team can use to assess performance. The KPIS generated include; total new users, total converted sales, and conversion rate. With the growing rate of data on the global scale there is an increasing demand for automation to improve efficiency, accuracy and reduce redundancy. This paper aims at providing a clear description/ procedure on how companies can enhance their business intelligence by leveraging KPI automation to save time and improve accuracy.

## Affidavit

I hereby declare in my word of honour

* that I prepared this final thesis independently and without any outside help.
* that I used no sources or resources other than those indicated.
* that I marked any verbatim quotes and paraphrased text by other authors within the work where they appear.
* that I did not submit it elsewhere for examination purposes.

I am fully aware that a false declaration will have legal consequences.

Ich erkläre hiermit ehrenwörtlich,

* diese Master-/Bachelor-/Seminararbeit selbstständig und ohne fremde Hilfe angefertigt,
* keine anderen als die angegebenen Quellen und Hilfsmittel benutzt,
* die Übernahme wörtlicher und sinngemäßer Zitate aus der Literatur an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet,
* die Arbeit mit gleichem Inhalt bzw. in wesentlichen Teilen noch nicht anderweitig für Prüfungszwecke vorgelegt zu haben.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

…….…….….…….…….…….…….…….…….…….…….… …….…….….…….…….…….…….…….…….…….…….…

Ort, Datum Unterschrift

The English text in this document only serves the purpose of providing information   
on the contents of the corresponding German text.

Only the German version of this affidavit is legally binding.

# Bibliography

|  |  |
| --- | --- |
| [1] | "Glosary," awork , [Online]. Available: https://www.awork.com/glossary/key-figures-or-performance-indicators#:~:text=Key%20performance%20indicators%20(KPIs)%20are,in%20taking%20measures%20for%20improvement.. |
| [2] | "Qlik," [Online]. Available: https://www.qlik.com/us/kpi#:~:text=KPIs%20provide%20targets%20for%20teams,forward%20at%20the%20strategic%20level.. |
| [3] | L. Muzamani, "IPC," 03 09 2024. [Online]. Available: https://www.ipcconsultants.com/blog/the-importance-of-automating-kpis-from-data-warehousing-power-bi-reporting-with-power-automate. |
| [4] | E. Buturla, "Monterail," [Online]. Available: https://www.monterail.com/blog/python-task-automation-examples. |
| [5] | "KPI.Org," KPI.Org, [Online]. Available: https://www.kpi.org/kpi-basics/#:~:text=Key%20Performance%20Indicators%20(KPIs)%20are,attention%20on%20what%20matters%20most.. |
| [6] | S. Penchikala, "InfoQ," 7 May 2025. [Online]. Available: https://www.infoq.com/news/2019/03/autonomous-analytics-bi/#:~:text=Autonomous%20analytics%20will%20help%20with,out%20of%20your%20investments?).. |
| [7] | "I," 24 Oct 2023. [Online]. Available: https://www.intrafocus.com/2023/10/the-challenges-of-kpi-management/. |
| [8] | "MadeToMeasure KPIs," MadeToMeasure KPIs , [Online]. Available: https://madetomeasurekpis.com/common-kpi-metric-reports-problems/#:~:text=Have%20you%20mapped%20out%20the,meetings%20still%20hadn't%20resumed!. |
| [9] | "MOOC," MOOC, 10 Feb 2022. [Online]. Available: https://www.mooc.org/blog/how-important-is-excel-in-business#:~:text=In%202019%2C%20market%20research%20indicated,standard%20for%20daily%20business%20analytics.. |
| [10] | G. Maksimenko, "Adexin," Adexin, 30 Oct 2024. [Online]. Available: https://adexin.com/blog/excel-big-data-limitations/. |
| [11] | S. Saci, "Medium," 7 September 2022. [Online]. Available: https://medium.com/data-science/automate-operational-reports-distribution-in-html-emails-using-python-c65c66fc99a6. |
| [12] | A. Shapiro, "Macrosoft," Macrosoft , May 2025. [Online]. Available: https://www.macrosoftinc.com/measuring-success-kpis-for-your-process-automation-initiatives/#:~:text=Use%20analytics%20and%20business%20intelligence,Employee%20Productivity%20&%20Satisfaction. |
| [13] | Algoschack, "Medium," Medium , 11 Dec 2023. [Online]. Available: https://medium.com/@algoshackt/the-positive-impact-of-python-test-automation-on-software-48ee3cef4475. |
| [14] | "python pptx," python pptx , [Online]. Available: https://python-pptx.readthedocs.io/en/latest/. |
| [15] | python, "python," python , [Online]. Available: https://www.python.org/doc/essays/blurb/. |
| [16] | "Scaling python," Scaling python, [Online]. Available: https://powerfulpython.com/blog/scaling-python/. |
| [17] | "Clover Dx," July 2023. [Online]. Available: https://www.cloverdx.com/blog/5-benefits-of-implementing-an-automated-reporting-process-in-your-organization. |
| [18] | "Inciper," Inciper , 2024. [Online]. Available: https://www.inciper.com/blog/5-reasons-to-begin-report-automation. |
| [19] | T. Stobierski, "Harvard," Harvard , [Online]. Available: https://online.hbs.edu/blog/post/data-driven-decision-making. |
| [20] | A. Nanji, "MarketingProfs," MarketingProfs, [Online]. Available: https://www.marketingprofs.com/charts/2023/50495/data-related-activities-marketers-spend-the-most-time-on. |
| [21] | "AI 2SQL," [Online]. Available: https://ai2sql.io/the-integration-of-sql-and-python. |
| [22] | "Martech," Martech , [Online]. Available: https://metricswatch.com/reasons-to-start-automated-reporting. |
| [23] | A. Kleiman, "E-Tabs," E-Tabs, Oct 2024. [Online]. Available: https://www.e-tabs.com/newsevents/powerpoint-report-automation-the-essential-guide/. |
| [24] | A. G. Alexandra Jonker, "IBM," 07 February 2025. [Online]. Available: https://www.ibm.com/think/topics/structured-vs-unstructured-data. |
| [25] | O. Pereverzieva, "Alexsoft," 16 December 2024. [Online]. Available: https://www.altexsoft.com/blog/structured-unstructured-data/. |
| [26] | J. Gulati, "KD Nuggests," September 2024 . [Online]. Available: https://www.kdnuggets.com/scalability-challenges-strategies-in-data-science. |
| [27] | O. Pereverzieva, "Altexsoft," 16 Dec 2024. [Online]. Available: https://www.altexsoft.com/blog/structured-unstructured-data/. |
| [28] | "wonder," 26th February 2021. [Online]. Available: https://askwonder.com/research/number-google-sheets-users-worldwide-eoskdoxav. |
| [29] | M. Sobolevskiy, "DataGrip," 23 December 2015. [Online]. Available: https://blog.jetbrains.com/datagrip/2015/12/23/how-many-sql-developers-is-out-there-a-jetbrains-report/#:~:text=According%20to%20Evans%20Data%20Corporation,of%20people%20using%20SQL%20today.. |
| [30] | M. Schacklett, "Datamation," 3 November 2023. [Online]. Available: https://www.datamation.com/big-data/structured-vs-unstructured-data/#:~:text=objects%20in%20searches.-,How%20do%20Companies%20use%20Structured%20and%20Unstructured%20Data?,into%20analytics%20and%20AI%20systems.. |
| [31] | pandas, "pandas," 2025. [Online]. Available: https://pandas.pydata.org/docs/getting\_started/intro\_tutorials/02\_read\_write.html. |
| [32] | "Talonic," Talonic GmbH, [Online]. Available: https://www.talonic.com/blog/how-startups-save-hours-by-automating-pdf-data-extraction#:~:text=The%20manual%20input%20was%20overwhelming,and%20significantly%20improved%20data%20accuracy.. |
| [33] | K. Bartley, "Rivery," 28 May 2025. [Online]. Available: https://rivery.io/blog/big-data-statistics-how-much-data-is-there-in-the-world/. |
| [34] | "Brief history on KPIs," Chief of staff network , [Online]. Available: https://www.chiefofstaff.network/blog/a-brief-history-of-okrs-and-kpis#:~:text=The%20concept%20of%20KPIs%20dates,ensure%20consistency%20and%20improve%20quality.. |
| [35] | "Unalighned new letter," 24 December 2024. [Online]. Available: https://www.unaligned.io/p/the-power-of-ai-in-business-intelligence. |
| [36] | 5 Dec 2024. [Online]. Available: https://www.gooddata.com/blog/service-as-a-software-is-the-new-saas-model/. |
| [37] | I. Cronin, "Unaligned News Letter," 24 December 2024. [Online]. Available: https://www.unaligned.io/p/the-power-of-ai-in-business-intelligence. |
| [38] | A. Undru, "ThoughtSpot," 8 May 2025 . [Online]. Available: https://www.thoughtspot.com/data-trends/analytics/retail-analytics. |
| [39] | A. Parashar, "code360," 10 March 2025. [Online]. Available: https://www.naukri.com/code360/library/python-libraries. |
| [40] | L. York, "UiPath," 22 September 2021. [Online]. Available: https://www.uipath.com/blog/automation/how-automation-unlocks-bi-analytics-full-potential#:~:text=Automation%20can%20aid%20the%20democratization,level%20that%20must%20be%20resolved.. |
| [41] | N. Rajput, "Medium," 31 May 2025. [Online]. Available: https://medium.com/@hadiyolworld007/the-role-of-python-in-building-autonomous-agents-powering-the-next-generation-of-smart-systems-392bec18689b. |
| [42] | M. Carter, "ShareArchiever," 28 December 2023. [Online]. Available: https://sharearchiver.com/blog/open-source-proprietary-data-analysys-tools/. |
| [43] | "Python central," 14 January 2025. [Online]. Available: https://www.pythoncentral.io/how-python-simplifies-api-development-a-guide-for-scalable-web-solutions/#:~:text=APIs%20are%20a%20core%20component,businesses%20to%20grow%20without%20bottlenecks.. |
| [44] | "Automation panda," [Online]. Available: https://automationpanda.com/2018/07/26/why-python-is-great-for-test-automation/. |
| [45] | "python tutorial," python tutorial , [Online]. Available: https://www.geeksforgeeks.org/python/best-python-modules-for-automation/. |
| [46] | "KD Nuggets," KD Nuggets , 19 November 2024. [Online]. Available: https://www.kdnuggets.com/10-python-libraries-every-data-analyst-should-know. |
| [47] | "Global Tech Council," 18 Feb 2025. [Online]. Available: https://www.globaltechcouncil.org/python/python-popularity-and-versatility/#:~:text=The%20language's%20extensive%20library%20ecosystem,the%20programming%20world%E2%80%8B%E2%80%8B.. |
| [48] | "apriorit," apriorit , 17 April 2023. [Online]. Available: https://www.apriorit.com/dev-blog/web-python-libraries-for-database-management. |
| [49] | "GeeksforGeeks," 12 June 2024 . [Online]. Available: https://www.geeksforgeeks.org/data-science/7-common-data-science-challenges-and-effective-solutions/. |
| [50] | 25 July 2023. [Online]. Available: https://aicadata.com/10-consequences-of-poor-data-quality-in-business-operations/#:~:text=The%20Solution:%20Data%20Cleansing,streamline%20and%20automate%20this%20process.. |
| [51] | "Fortra," Fortra , 7 April 2025. [Online]. Available: https://dataclassification.fortra.com/blog/data-anonymization-techniques-protecting-privacy-data-sets#:~:text=Data%20anonymization%20is%20the%20mechanism%20used%20to,insights%20without%20violating%20privacy%20regulations%20and%20norms.. |
| [52] | "Medium," Medium , 27 August 2023. [Online]. Available: https://anshsingh2002.medium.com/challenges-faced-by-data-analysts-while-performing-data-preprocessing-3724d8cd7ba0. |
| [53] | "Datacamp," Datacamp , 15 Jan 2025. [Online]. Available: https://www.datacamp.com/blog/data-preprocessing. |
| [54] | "GeekforGeeks," 25 August 2024. [Online]. Available: https://www.geeksforgeeks.org/data-engineering/real-time-data-processing-challenges-and-solutions-for-streaming-data/. |
| [55] | "DataVersity," 1 March 2024. [Online]. Available: https://www.dataversity.net/the-impact-of-poor-data-quality-and-how-to-fix-it/. |
| [56] | "Vorecol," [Online]. Available: https://blogs.psico-smart.com/blog-what-role-do-key-performance-indicators-kpis-play-in-measuring-organizational-performance-15418. |
| [57] | "KPI report," Qlick , [Online]. Available: https://www.qlik.com/us/kpi/kpi-reporting. |
| [58] | "Key Automation statistics," vena, 15 Jan 2025. [Online]. Available: https://www.venasolutions.com/blog/automation-statistics#:~:text=Key%20Automation%20Statistics%20at%20a,The%20Rise%20of%20Business%20Automation. |
| [59] | "Financier," Jan 2021. [Online]. Available: https://www.financierworldwide.com/accelerating-digital-transformation-in-the-new-normal. |
| [60] | "DOMO," 9 January 2025. [Online]. Available: https://www.domo.com/learn/article/kpi-tracking#:~:text=KPI%20tracking%20helps%20your%20organization%20develop%20a,use%20data%20to%20drive%20strategies%20and%20actions.. |
| [61] | A. Smith, "wasabi," 15 August 2022 . [Online]. Available: https://wasabi.com/blog/data-management/cloud-object-storage-zettabytes. |
| [62] | "row zero," 3 January 2025. [Online]. Available: https://rowzero.io/blog/excel-row-limit. |
| [63] | C. Mullins, "Tech Target," 19 June 2020. [Online]. Available: https://www.techtarget.com/searchdatamanagement/definition/data-aggregation#:~:text=Data%20aggregation%20is%20any%20process,query%20large%20sets%20of%20data.. |
| [64] | "Plotly dash," Plotly , [Online]. Available: https://dash.plotly.com/tutorial. |
| [65] | "Kaggle," 2024. [Online]. Available: https://www.kaggle.com/datasets/jyotidabas/online-sales-data. |
| [66] | "Ryte," Demographic data, [Online]. Available: https://en.ryte.com/wiki/Demographic\_Data/. |
| [67] | "CueMath," [Online]. Available: https://www.cuemath.com/summation-formulas/. |
| [68] | "Google ads help," Google ad , [Online]. Available: https://support.google.com/google-ads/answer/2684489?hl=en#:~:text=Conversion%20rates%20are%20calculated%20by%20simply%20taking,5%25%2C%20since%2050%20%C3%B7%201%2C000%20=%205%25.. |
| [69] | G. Adekoya, "Medium," Importing python libraries in VS code, 13 August 2023. [Online]. Available: https://medium.com/@gbemiadekoya/importing-python-libraries-in-vs-code-e9e7806586a7. |
| [70] | "pandas," pandas , 2025. [Online]. Available: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.empty.html. |
| [71] | "aws," aws, [Online]. Available: https://aws.amazon.com/what-is/structured-data/#:~:text=Structured%20data%20is%20data%20that,is%20more%20challenging%20to%20analyze.. |
| [72] | "pandas," pandas , 2025. [Online]. Available: https://pandas.pydata.org/docs/getting\_started/intro\_tutorials/02\_read\_write.html. |
| [73] | "pandas," pandascut , 2025. [Online]. Available: https://www.python4data.science/en/latest/workspace/pandas/discretisation.html. |
| [74] | "python pptx," python pptx , 2025. [Online]. Available: https://python-pptx.readthedocs.io/en/latest/dev/analysis/dml-fill.html. |
| [75] | "Vultr," Vultr , 25 December 2024. [Online]. Available: https://docs.vultr.com/python/third-party/pandas/DataFrame/sum. |
| [76] | "GeekforGeeks," GeekforGeeks, 10 Dec 2024. [Online]. Available: https://www.geeksforgeeks.org/bar-plot-in-matplotlib/. |
| [77] | "GeekforGeeks," GeekforGeeks, 02 May 2025. [Online]. Available: https://www.geeksforgeeks.org/python/creating-and-updating-powerpoint-presentations-in-python-using-python-pptx/. |
| [78] | "GeeksforGeeks," Python Tutorial , 02 May 2025. [Online]. Available: https://www.geeksforgeeks.org/python/creating-and-updating-powerpoint-presentations-in-python-using-python-pptx/. |
| [79] | "Gitlab," python pptx , May 2019. [Online]. Available: https://github.com/scanny/python-pptx/issues/533. |
| [80] | "Plotly Dash," Plotly , [Online]. Available: https://dash.plotly.com/tutorial. |
| [81] | "Medium," Medium, 19 April 2024. [Online]. Available: https://medium.com/plotly/how-to-create-a-beautiful-interactive-dashboard-layout-in-python-with-plotly-dash-a45c57bb2f3c. |
| [82] | "Dash Plotly," Dash , [Online]. Available: https://dash.plotly.com/basic-callbacks. |
| [83] | "Dash Plotly," Dash, [Online]. Available: https://dash.plotly.com/tutorial. |
| [84] | Marcelo, "Logrise," Logrise, 18 Oct 2021. [Online]. Available: https://logrise.com/en/9-reasons-to-automate-kpis/#reduzier-tempo-nos-kpi. |