1. Secure data access: You should ensure that access to sensitive data is limited to only authorized personnel. This can be achieved through the use of access controls and user authentication mechanisms. For example, you can set up role-based access controls, where users are only granted access to the data they need to perform their job functions.
2. Data encryption: You can encrypt sensitive data to protect it from unauthorized access. Encryption involves transforming data into a code that can only be deciphered with a key. This can help protect the data from hackers or unauthorized access in case of a security breach.
3. Regular data backups: Backing up data regularly can help ensure that data can be recovered in the event of a security breach or other data loss event. It's important to have a backup and disaster recovery plan in place and to test it regularly to ensure its effectiveness.
4. Data anonymization: In some cases, it may be necessary to share data with third parties for analysis or research purposes. In these situations, it's important to ensure that the data is anonymized, so that individuals cannot be identified from the data. This can help protect the privacy of individuals and ensure compliance with GDPR and other data protection regulations.
5. Secure data storage: You should ensure that sensitive data is stored securely, using appropriate physical and logical security measures. This can include measures such as firewalls, antivirus software, and data encryption.
6. Data retention policies: You should ensure that data is retained only for as long as necessary and is disposed of securely when it is no longer needed. This can help reduce the risk of data breaches or other security incidents.

!pip install pdfplumber

import pandas as pd

import pdfplumber

# Open the PDF file in read-binary mode

with pdfplumber.open('input\_file.pdf') as pdf\_file:

# Get the total number of pages in the PDF file

total\_pages = len(pdf\_file.pages)

# Initialize empty list to store extracted data

data = []

# Loop through each page of the PDF file

for page\_num in range(total\_pages):

# Get the page object

page = pdf\_file.pages[page\_num]

# Extract the text from the page

page\_text = page.extract\_text()

# Split the text into lines

lines = page\_text.split('\n')

# Loop through each line of the page

for line in lines:

# Split the line into columns using delimiter

columns = line.split(',')

# Append the columns to the data list

data.append(columns)

# Convert the data list to a DataFrame

df = pd.DataFrame(data)

# Save the DataFrame to an Excel file

df.to\_excel('output\_file.xlsx', index=False)

!pip install PyMuPDF

import pandas as pd

import fitz

# Open the PDF file

with fitz.open('input\_file.pdf') as pdf\_file:

# Get the total number of pages in the PDF file

total\_pages = pdf\_file.page\_count

# Initialize empty list to store extracted data

data = []

# Loop through each page of the PDF file

for page\_num in range(total\_pages):

# Get the page object

page = pdf\_file[page\_num]

# Extract the text from the page

page\_text = page.get\_text()

# Split the text into lines

lines = page\_text.split('\n')

# Loop through each line of the page

for line in lines:

# Split the line into columns using delimiter

columns = line.split(',')

# Append the columns to the data list

data.append(columns)

# Convert the data list to a DataFrame

df = pd.DataFrame(data)

# Save the DataFrame to an Excel file

df.to\_excel('output\_file.xlsx', index=False)

!pip install camelot-py

import pandas as pd

import camelot

# Extract tables from PDF file

tables = camelot.read\_pdf('input\_file.pdf')

# Convert each table to a DataFrame and append to list

dfs = []

for table in tables:

df = table.df

dfs.append(df)

# Concatenate DataFrames into single DataFrame

df = pd.concat(dfs)

# Save the DataFrame to an Excel file

df.to\_excel('output\_file.xlsx', index=False)

pip install pdfminer.six

import pandas as pd

from pdfminer.high\_level import extract\_text

# Extract text from PDF file

pdf\_text = extract\_text('input\_file.pdf')

# Split text into lines

lines = pdf\_text.split('\n')

# Initialize empty list to store extracted data

data = []

# Loop through each line of text

for line in lines:

# Split the line into columns using delimiter

columns = line.split(',')

# Append the columns to the data list

data.append(columns)

# Convert the data list to a DataFrame

df = pd.DataFrame(data)

# Save the DataFrame to an Excel file

df.to\_excel('output\_file.xlsx', index=False)