```python
import os
import boto3
import zipfile
import io
import logging
import pandas as pd
from datetime import datetime

logger = logging.getLogger()
logger.setLevel(logging.INFO)

s3_client = boto3.client('s3')

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", event)

        # Get the S3 bucket and object key from the event
        bucket = event['Records'][0]['s3']['bucket']['name']

        key = event['Records'][0]['s3']['object']['key']

        RX_file_name = os.path.basename(key)


        #getting date from file name
        RX_file_date = RX_file_name.split("_")[-1].split(".")[0]

        # Parse the date
        date_obj = datetime.strptime(RX_file_date, "%m%d%Y")

        # Format the date as yyyy/mm/dd
        formatted_date = date_obj.strftime("%Y/%m/%d")

        # Download the zip file from S3
        response = s3_client.get_object(Bucket=bucket, Key=key)
        zip_data = response['Body'].read()

        # Extract the zip file
        with zipfile.ZipFile(io.BytesIO(zip_data)) as zip_ref:
            for file_name in zip_ref.namelist():
                if file_name.endswith(".RRF"):
                    # Extract the file
                    with zip_ref.open(file_name) as file_data:
                        df = pd.read_csv(file_data, delimiter='|', header=None)
                        df=df.iloc[:,:-1]
                        df["code_set"] = RX_file_name.split("_")[0]
                        df["Version Month"] = formatted_date

                    # Get header names from the corresponding Excel sheet
                    num_cols_df = df.shape[1]
                    sheet_name = os.path.splitext(os.path.basename(file_name))[0]  # Assuming file names match sheet names
                    header_file_key = 'excelfile/RxNorm_Header.xlsx'               # Change to your actual file name
                    header_df = read_excel_from_s3(bucket, header_file_key, sheet_name)
                    if header_df is not None:
                        df.columns = header_df.iloc[:, 0].tolist()  # Assuming headers are in column B

                    # Convert DataFrame to CSV format
                    csv_data = df.to_csv(index=False)

                    # Upload the CSV file to another folder in the same S3 bucket
                    destination_key = f"rxfiles/{os.path.basename(file_name.replace(".RRF",".csv"))}"
                    s3_client.put_object(Bucket=bucket, Key=destination_key, Body=csv_data)

        return {
            'statusCode': 200,
            'body': 'Extraction and file upload completed'
        }
    except Exception as e:
        logger.error("Error: %s", e, exc_info=True)
        return {
            'statusCode': 500,
            'body': f'Error: {e}'
        }
```

```python
def read_excel_from_s3(bucket, key, sheet_name):
    try:
        response = s3_client.get_object(Bucket=bucket, Key=key)
        excel_bytes = response['Body'].read()
        df = pd.read_excel(io.BytesIO(excel_bytes), sheet_name=sheet_name, header=None)
        return df
    except Exception as e:
        logger.error("Error reading Excel file: %s", e)
        return None
```