

**Project Title**: **Automated Sales Forecasting and Data Pipeline using AWS S3 and Machine Learning**

### ****Project Overview****

This project aims to build a fully automated pipeline for daily sales data updates, preprocessing, forecasting, and reporting. By leveraging **AWS S3**, **Python**, and machine learning, the project will automate data ingestion, cleaning, predictive analysis, and reporting.

1. Automate Daily Data Update to S3

* Set up a system (could be a script or an API) that pulls daily sales data from the source (e.g., databases, CSVs, or other sources) and uploads it to S3 at the end of each day.
* Use AWS Lambda or a scheduled task (cron job) to automate the process.
* Ensure the data is saved with a unique timestamp or file name for easy tracking (e.g., sales\_data\_2024\_12\_13.xlsx).

**Deliverables**:

* Automated script or Lambda function that uploads the daily data.
* Confirmation of daily upload with timestamp

**Step 2: Pull the Latest Data from S3**

* Write a Python script to connect to S3 using Boto3 and download the latest file.
* Ensure that the script checks the S3 bucket for the most recent file based on the timestamp or file naming convention.
* Download the latest data file for processing.

**Deliverables**:

* Python script to fetch the latest file from S3.
* Output showing the successful download of the data file.

**Step 3: Data Preprocessing and Cleaning**

**Objective**: clean and prepare the data for analysis.

**Instructions**:

1. Load the latest file into Pandas.
2. Check for missing values and clean them (imputation or removal).
3. Convert date columns to datetime format, if not already done.
4. Aggregate sales data by the day, week, or month, depending on the granularity of the prediction.
5. Handle any outliers or anomalies that might affect the analysis.

**Deliverables**:

* Python script for preprocessing the downloaded data.
* Cleaned dataset output (with missing values handled, aggregated data, etc.).

**Step 4: Exploratory Data Analysis (EDA)**

**Objective**: Explore the data to identify trends and patterns.

**Instructions**:

1. Visualize the sales and inventory data (e.g., daily sales trends, inventory stock levels).
2. Check for seasonality or trends over time (e.g., using time series analysis).
3. Calculate basic statistics (mean, median, standard deviation) and discuss insights from the data.
4. Identify any possible external factors affecting the sales or inventory (e.g., holidays, promotions).

**Deliverables**:

* Python script with visualizations (using Matplotlib/Seaborn).
* Insights and interpretation of the data trends.

**Step 5: Feature Engineering**

**Objective**: create features needed for prediction models.

**Instructions**:

1. Derive features such as:
   * Day of the week
   * Month of the year
   * Moving averages (e.g., 7-day rolling average)
   * Lag features (e.g., sales from the previous day/week/month/quarter)
   * Seasonal indicators (e.g., holiday season, weekends)
2. Ensure that the features are ready for use in a machine learning model.

**Deliverables**:

* Python script to create new features.

**Step 6: Model Training and Prediction**

**Objective**: train a machine learning model to predict future sales.

**Instructions**:

1. Split the data into training and testing sets.
2. Train a regression model (e.g., Random Forest, ARIMA, or Linear Regression) to predict the next day's or next week's sales.next year,next month sales
3. Evaluate the model performance using appropriate metrics (e.g., Mean Squared Error).
4. **Use the model to predict future sales for next year (weekly/monthly/quarterly)**

**Deliverables**:

* Python script to train and test the model.
* Model evaluation results (e.g., MSE).
* Predicted sales for the next period.

**Step 7: Daily Prediction and Updating Process**

**Objective**: Automate the prediction process daily using new data.

**Instructions**:

1. Set up a cron job or a Lambda function that runs the entire pipeline daily, starting from pulling the latest data from S3 to making predictions.
2. Ensure the predictions are updated and stored (e.g., in a CSV or database).
3. Create a log of predictions to monitor performance over time.

**Deliverables**:

* Python script or Lambda function to automate the prediction process.
* Daily prediction logs and updated predictions.

**Step 8: Save Predictions and Upload to S3**

**Objective**: save their predictions and upload them back to S3.

**Instructions**:

1. After generating the predictions, save them into a file (e.g., CSV or Excel).
2. Upload the prediction file back to S3 for future reference and analysis.
3. Ensure the file name contains the date of the prediction (e.g., predictions\_2024\_12\_14.csv).

**Deliverables**:

* Python script to save and upload predictions to S3.
* Confirmation of successful upload (with file name and timestamp).

**Step 9: Reporting and Visualization**

**Objective**: create reports to visualize the results of their analysis.

**Instructions**:

1. Generate a final report that includes:
   * Sales trends and prediction results.
   * Accuracy of predictions.
   * Business insights (e.g., how predictions can help in inventory management, demand forecasting).
2. Visualize the results using charts and graphs.
3. Prepare a summary of the project and present it (e.g., in a PowerPoint or Jupyter Notebook).

**Deliverables**:

* Final report with visuals.
* Presentation summarizing the analysis, predictions, and insights.