REV LEASING SCORE℠

Technical Assessment: Automation of Follow-Up Tracking

Premise:
Rev conducts mystery shop assessments on multifamily properties to simulate real traffic and interactions with leasing agents. As part of this process, mystery shoppers are given aliases, which consist of fake names, emails, and phone numbers. These aliases are used during conversations with the property teams. Over a 14-day period, multiple contacts are made with each property to assess their response time, follow-up efforts, and overall engagement.
At the end of the 14-day assessment period, properties will often follow up with the aliases via email, text messages, or voicemails. The challenge is to track and accurately identify which property sent each follow-up and which alias received it. Since follow-up tracking is only done once at the end of the assessment period, it is crucial to segregate and classify all follow-up communications correctly.
The purpose of this technical assessment is to create an automated system that can identify and categorize all follow-up communications (emails, texts, and voicemails) after the assessments are completed. This will ensure that each follow-up is matched with the correct property and alias, and a consolidated report is generated for review.

Objective:
Build a Python automation script that:

- Collects all follow-up emails, text messages, and voicemails from properties after the 14-day mystery shop assessment period.
- Matches these communications to the correct property and alias.
- Generates a detailed report of follow-ups for each property-alias combination.

The script will take the following input files:

1. **Alias Email login credentials** (for accessing emails sent to the aliases).
2. **Text message and voicemail logs** (exported from the VoIP tool).
3. **Property details** (Property name, ID, phone number).
4. **Alias details** (Alias name, email, phone number).

The output will be a **final follow-up report** that provides details of each communication along with the text logs.

Task 1: Download Emails from Alias Accounts
Goal: Retrieve all incoming emails sent to alias email accounts.

1. **Email Integration:**
    - Use an appropriate Python email library (e.g., IMAPlib, Gmail API) to connect to each alias email account using the credentials provided in the input files.
    - Authenticate using email addresses and passwords for the aliases.
2. **Fetch Emails:**
    - Download all incoming emails from the inbox (and optionally spam).
    - Extract relevant details such as the sender's email, subject, body, and any attachments.
3. **Store Email Data:**

- Save the extracted email information (sender, subject, body, attachments) in a structured format (CSV or JSON).

Deliverable: A Python script that retrieves and stores inbound email data from alias email accounts.

Task 2: Identify Property from Emails

Goal: Determine which property each email was sent from.

1. **Load Property Data:**
   - Import the property details file (Property name, ID, phone number).
2. **Analyze Email Content:**
   - Check the sender's email address, subject, and body for references to the property (e.g., property name, phone number, etc.).
   - Match the email to the correct property based on the sender's details or any property-related keywords in the email.
3. **Map Emails to Properties:**
   - Assign the correct Property ID and Alias ID based on the email content.

Deliverable: A file mapping each email to its corresponding property and alias.

Task 3: Process Text Message and Voicemail Logs

Goal: Analyze text messages and voicemail logs to identify which property sent them and to which alias they were sent.

1. **Load Logs:**
   - Import the text message and voicemail logs from the VoIP tool. These logs will contain the sender and receiver's phone numbers as well as the message content or voicemail transcript.
2. **Match Phone Numbers:**
   - Cross-reference the phone numbers in the logs with those in the alias and property details.
   - If the sender's phone number is not found, analyze the message content or voicemail transcript for property names (Use natural language processing (NLP) techniques or string matching to identify properties and aliases mentioned in the text messages or voicemail transcripts.)
3. **Map Communications to Properties:**
   - Assign each text message or voicemail to the correct Property ID and Alias ID.

Deliverable: A file mapping each text message and voicemail to its corresponding property and alias.

Task 4: Generate Final Follow-Up Report

Goal: Combine the results from Tasks 1–3 and produce a final report.

1. **Consolidate Data:**
   - Merge all the follow-up data (emails, text messages, voicemails) into one unified report.
2. **Generate Summary:**

- Summarize the total number of follow-ups received for each property and alias.
- Highlight any properties that did not send follow-up communications.

Deliverable: A final report that lists all follow-ups, including:

- Property ID
- Alias ID
- Type of communication (email, text, voicemail)
- Date and time of communication
- A summary of the content or transcript of the follow-up

Task 5: Classify Follow-Ups

Goal: Classify each follow-up into specific categories based on its content.

1. **Analyze Follow-Up Content:**
   - Identify whether the communication includes a tour confirmation, booking link, request to schedule a tour, or any attachments such as pictures.
2. **Personalization Check:**
   - Determine whether the follow-up was personalized (mentions the alias) or a general follow-up.

Deliverable: The final output should include the following tags for each follow-up:

- **Tour Confirmation** (Yes/No)
- **Booking Link** (Yes/No)
- **Requests Tour Booking** (Yes/No)
- **Contains Pictures** (Yes/No)
- **Personalized** or **Generalized** follow-up

Task 6: Store and Host Data in SQL Platform

Goal: Host all raw and processed follow-up data on a SQL platform for easy querying and access.

1. **SQL Database Setup:**
   - Set up a SQL database (e.g., MySQL, PostgreSQL, or SQLite) to host both raw and processed data.
   - Create tables to store:
     - Alias details (Alias ID, email, phone number).
     - Property details (Property ID, name, phone number).
     - Follow-up communications (type, date/time, content, and classification tags).
     - Logs (email logs, text message logs, voicemail logs).
2. **Data Insertion:**
   - Modify the Python script to:
     - Insert raw follow-up data (emails, texts, and voicemails) into the appropriate tables.
     - Insert processed and classified data into relevant tables after analysis is complete.
3. **Data Querying:**

    - Set up basic SQL queries to retrieve:
        - All follow-up communications for a specific property.
        - Summaries of follow-ups (e.g., number of personalized vs. generalized communications, tour confirmations).
        - Properties with no follow-ups.
4. **Ensure Data Integrity:**
    - Implement error handling to avoid inserting duplicate or incorrect data.
    - Create indexes on frequently queried fields (e.g., Property ID, Alias ID) for faster lookups.

Deliverable:

- A SQL database that stores all raw and processed follow-up data and can be queried to generate reports and summaries.
- Python script modified to integrate SQL operations (data insertion and querying).

Summary:

The entire technical assessment should be completed as **one single Python script**. The script will take the input files (Alias email credentials, text message and voicemail logs, Property details, Alias details) and output a final consolidated follow-up report. The goal is to accurately identify which property each follow-up came from, which alias received it, and classify the type of communication for each follow-up.