Assessment Task: Custom Email Sender

Internship Details:

Stipend: €200-400 per month, based on experience and performance
Duration: 3 months
Location: Remote

Project Deadline and Submission Details

- **Deadline**: November 18th 2024
- **Submission**: Please send your completed project, including the GitHub repository link, to **kapil@breakoutinvesting.in**

Welcome to the assessment! This document provides a complete overview of the requirements and expected results, along with answers to potential questions. This project will help us evaluate your skills in machine learning, API integration, and prompt engineering, as well as your approach to creating user-friendly applications.

Objective

Create a custom email-sending application with a front-end dashboard that performs the following tasks:

1. Reads data from a Google Sheet or CSV file for email customization.
2. Allows users to connect their email account.
3. Accepts a customizable prompt for personalizing each email.
4. Customizes and sends emails using an LLM or other content-generation approach.
5. Displays real-time status and analytics for sent emails.
6. Provides email scheduling, throttling, and delivery tracking using an Email Service Provider (ESP) integration.

Candidates may use any tools, libraries, or frameworks of their choice for this task.

Assessment Task: Custom Email Sender

Detailed Requirements

1. Data Connection

- **Input Data**:
  - Allow users to connect a Google Sheet or upload a CSV file containing email data with columns. For eg a sheet could contain data containing like `Company Name`, `Location`, `Email`, and `Products`. (This is just for reference)

---

2. Email Integration

- **Account Connection**:
  - Provide an option for users to connect their email account for sending emails using either a personal account (e.g., Gmail, Outlook) or a configured ESP.
- **Implementation Tips**:
  - Use OAuth2 for secure email service connections or provide configurations for SMTP/ESP integration.

---

3. Customizable Prompt Box for Email Content

- **Prompt Box**:
  - Allow users to input a customizable prompt with placeholders, such as `{Company Name}` or `{Location}`, which will be replaced by actual values from each row in the dataset.

---

4. Column Detection and Dynamic Field Replacement

- **Auto-detect Columns**:
  - Automatically detect columns in the dataset and allow users to insert these fields as placeholders in the prompt box.

---

5. Email Customization and Sending

- **Content Generation**:
  - For each row, generate a custom message based on the user's prompt and row data. Use LLMs API to generate a response. (for ex. Groq API)
- **Email Sending**:
  - Customize the content for each recipient and send the email.

---

6. Email Scheduling and Throttling

- **Scheduling**:
  - Allow users to schedule emails for specific times. Provide options to schedule all emails for a specific time or stagger them over intervals.
  - If possible, allow setting different schedules for individual or batch emails (e.g., send 50 emails per hour).
- **Throttling**:
  - Provide the option to throttle email sending to stay within provider limits (e.g., send X emails per minute or hour).
  - Include a user-configurable rate-limiting option.
- **Implementation Tips**:
  - Store scheduling data in a database or use in-memory scheduling libraries.
  - Implement a queuing system, such as `Celery`, or use time-based functions in the chosen framework to manage scheduling and throttling.

---

7. Real-Time Analytics for Sent Emails

- **Analytics Dashboard**:
  - Provide real-time analytics on sent emails, such as:
    - **Total Emails Sent**
    - **Emails Pending**
    - **Emails Scheduled**
    - **Emails Failed**
    - **Response Rate (if available)**
- **Implementation Tips**:
  - Use a background job to update email status metrics regularly.
  - Display analytics using charts or counters on the dashboard, updating the data in real-time or at regular intervals.

---

8. Email Delivery Tracking with ESP Integration

- **ESP Integration**:
  - Integrate with an Email Service Provider (ESP) such as SendGrid, Amazon SES, or Mailgun to track email delivery.
  - Display delivery statuses, such as Delivered, Opened, Bounced, and Failed, on the dashboard.
- **Example ESPs**:
  - **SendGrid**: Use the SendGrid API for delivery tracking and to access event-based data.

  - **Amazon SES**: Set up Amazon SES notifications to track email delivery, bounces, and complaints.
  - **Mailgun**: Integrate with Mailgun's Events API to retrieve detailed delivery and engagement metrics.
- **Implementation Tips**:
  - Use the ESP's event webhooks to receive real-time updates for each email.
  - Store and update email delivery metrics (Delivered, Bounced, Opened) in a database or dictionary and reflect this in the dashboard.

---

9. Real-Time Dashboard for Email Status and Tracking

- **Dashboard**:
  - Display email send status (e.g., Sent, Scheduled, Pending, Failed) and track delivery status (Delivered, Opened, Bounced) using the ESP's tracking data.
  - Include a progress bar or live status indicator to show overall email sending progress.
- **Example Dashboard**:

Company Name	Email	Status	Delivery Status	Opened
ABC Corp	contact@abccorp.com	Sent	Delivered	Yes
XYZ Ltd	info@xyzltd.co.uk	Scheduled	N/A	N/A
DEF Inc	hello@definc.com	Failed	Bounced	No

- **Implementation Tips**:
  - Use front-end widgets for tracking progress and displaying real-time analytics.
  - Consider using WebSocket or polling to update the dashboard in real-time as email statuses change.

Assessment Task: Custom Email Sender

Technical Requirements

1. **Libraries and Frameworks**: Candidates may choose any libraries or frameworks they prefer.
2. **Storage**: Store scheduling information, email statuses, and analytics data in a database or appropriate in-memory storage.
3. **Documentation**: Provide a README with:
    - Setup and configuration instructions, including how to obtain and configure API keys for the chosen ESP.
    - Steps to configure email scheduling and throttling.
    - Usage instructions.

Evaluation Criteria

- **Functionality**: Does the solution meet all requirements, including email scheduling, tracking, and analytics?
- **Code Quality**: Is the code well-structured and documented?
- **User Experience**: Is the dashboard intuitive and user-friendly?
- **Reliability**: Does the solution handle errors and ESP limits gracefully?
- **Documentation**: Is the README clear and detailed?

Submission Guidelines

1. Submit your code via GitHub.
2. Include a `README.md` with setup and usage instructions.
3. Optionally, include a short video demonstrating the solution.

Stay Connected:

Follow **Kapil Mittal** on LinkedIn for updates and opportunities: LinkedIn Profile.

Join the Telegram channel for updates in trading and investment: Telegram @breakoutinvesting.

Check out other trading insights and strategies on TradingView: TradingView Profile.

Visit the Breakout AI website for more about what we do: breakoutai.tech.