CXINTEL – CUSTOMER EXPERIENCE INTELLIGENCE PLATFORM

Phase 5: Python AI Integration

```
from simple_salesforce import Salesforce
from textblob import TextBlob

# Connect to Salesforce
sf = Salesforce(
    username='your user name',
    password='your password',
    security_token='your security token',
    domain='login'  # or 'test' if sandbox
)

print(" Logged into Salesforce!")

# Sentiment Analysis Function
def analyze_sentiment(text):
    blob = TextBlob(text)
    score = round(blob.sentiment.polarity, 3)  # score between -1 and 1
    if score > 0.2:
        return "Positive", score
    elif score < -0.2:
        return "Negative", score
    else:
        return "Neutral", score

# Query Unprocessed Feedback Records
```

```
query = """
    SELECT Id, Feedback_Text__c
    FROM CustomerFeedback__c
    WHERE Sentiment__c = NULL
    LIMIT 50
"""

results = sf.query(query)
records = results['records']

print(f"Found {len(records)} records to process.")

# Process Each Record
for rec in records:
    feedback_id = rec['Id']
    feedback_text = rec.get('Feedback_Text__c', '')

    if not feedback_text:
        print(f"Skipping record {feedback_id} - no feedback text.")
        continue

    sentiment, score = analyze_sentiment(feedback_text)

    # Update record in Salesforce
    sf.CustomerFeedback__c.update(feedback_id, {
        'Sentiment__c': sentiment,
        'Sentiment_Score__c': score,
        'Processed__c' : True
    })
    print(f"Updated record {feedback_id} → {sentiment} ({score})")

print("All feedback records processed.")
```

- **Prerequisites Checklist**

Activate your virtual environment:

- **In VS Code terminal:**

  ```
  venv\Scripts\activate
  ```

- **Make sure the required libraries are installed:**

  ```
  pip install simple-salesforce textblob
  ```

- Confirm your Salesforce credentials (username, password, token) are correct and the fields Sentiment__c and Sentiment_Score__c exist on your Customer_Feedback__c object.

- **Double-check:** Your two records should have Sentiment__c = NULL and Feedback_Text__c filled in.

- **Now Run the Script**

  In the terminal:

  ```
  python sentiment_analyzer.py
  ```

- **What you should see:**

  ```
  Logged into Salesforce!
  
  Found 2 records to process.
  
  Updated record 001XXXXXXXX → Positive (0.35)
  
  All feedback records processed.
  ```

- **Scheduling the Python Script:**

- Since you're running the script from your Windows 11 system and it's an external Python script (not inside Salesforce), you can schedule it from your machine.

- **Note:** Salesforce cron jobs (like Apex Scheduler) only run Apex code inside Salesforce. To run external Python scripts, you'll use:

- **Windows Task Scheduler**

  How to Schedule Your Python Script on Windows 11

  What you need:

  You already have a Python script: sentiment_analyzer.py

  It works when you manually run: python sentiment_analyzer.py

  Your virtual environment is activated as: venv

Folder Setup (Recommended)

Project folder (example: C:\Users\YourName\Documents\sentiment-feedback)

```
├── venv
├── sentiment_analyzer.py
```

You will need to run the script using the python.exe from your venv folder.

➢ Step-by-Step: Schedule with Task Scheduler

- Open Task Scheduler

  Press Windows Key → Search "Task Scheduler" → Open it

  Click Create Basic Task…

  Name: Salesforce Sentiment Analyzer

  Description: Automatically processes Salesforce feedback with sentiment AI

  Click Next

- **Choose Trigger:**

  Daily or Weekly (your choice)

  Example: Daily → Click Next → Choose time: 09:00 AM

- **Choose Action:**

  Select Start a Program → Click Next

  Program/Script box:

- **Browse and select:**

  C:\Users\YourName\Documents\sentiment-feedback\venv\Scripts\python.exe

  (This runs Python from your virtual environment)

- **Add Arguments (optional):**

  sentiment_analyzer.py > output.log 2>&1

- **Start in (important):**

  C:\Users\YourName\Documents\sentiment-feedback

  (This is the directory where your script lives)

  Click Finish

➢ Deploy Python App to Cloud (Github)

✓ **Goal: Your Python script will run automatically every day/hour from the cloud via GitHub Actions.**

Step 0: Pre-requisites

Make sure you have:

- A free GitHub account
- Your Python script working locally (you do ✅)
- Git installed on your system (check with git --version)
- GitHub CLI or browser access

Step 1: Create a GitHub Repository

1. Visit: https://github.com
2. Click on New → Create a repository
   - Name: sentiment-scheduler
   - Description: Salesforce sentiment automation
   - Make it private or public (your choice)
   - Initialize with a README (optional)
3. Copy the repository URL for later (e.g. https://github.com/yourname/sentiment-scheduler)

Step 2: Prepare Your Project Locally

1. In VS Code terminal:

```
cd path\to\your\project
git init
git remote add origin https://github.com/yourname/sentiment-scheduler
git branch -M
```

2. Create these files:

sentiment_analyzer.py ← Your working Python script

- **requirements.txt :**

```
simple-salesforce
textblob
```

- **.gitignore :**
- venv/
  pycache/
  *.log

3. Add files to Git and push:

```
git add .
git commit -m "Initial commit"
git push -u origin main
```

Step 3: Store Salesforce Credentials Securely

1. Go to your GitHub repo → Settings → Secrets and variables → Actions → New repository secret

Add these:

- SF_USERNAME
- SF_PASSWORD
- SF_TOKEN

(Use your actual Salesforce username, password, and security token.)

Step 4: Add a GitHub Actions Workflow

1. In your repo, create a folder called:

   .github/workflows

2. Inside that, create a file named: run-sentiment.yml

   Paste this YAML:

```
name: Run Salesforce Sentiment Analyzer

on:

schedule:

 - cron: '0 9 * * *'  # runs daily at 9:00 AM UTC

workflow_dispatch:

jobs:

run-script:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v3

- name: Set up Python

uses: actions/setup-python@v4

with:
```

```
python-version: '3.10'

- name: Install dependencies

run: |

python -m pip install --upgrade pip

pip install -r requirements.txt

- name: Run script

env:

SF_USERNAME: ${{ secrets.SF_USERNAME }}

SF_PASSWORD: ${{ secrets.SF_PASSWORD }}

SF_TOKEN: ${{ secrets.SF_TOKEN }}

run: python sentiment_analyzer.py
```

Step 5: Modify sentiment_analyzer.py to use env variables

At the top of your script, add:

```
import os
```

Replace your Salesforce connection block with:

```
sf = Salesforce(
 username=os.environ['SF_USERNAME'],
 password=os.environ['SF_PASSWORD'],
 security_token=os.environ['SF_TOKEN'],
 domain='login'
 )
```

Step 6: Push Workflow to GitHub

In terminal:

```
 git add .
 git commit -m "Added GitHub Actions workflow"
 git push
```

Step 7: Run & Verify

1. Go to your GitHub repo → Actions tab
2. You'll see: "Run Salesforce Sentiment Analyzer" → Click "Run workflow"
3. Check logs → You'll see your script's output!

   Done! Your script will now run automatically every day at 9:00 AM UTC.