Withers-Project /
func-image-ingest

Code | Issues | Pull requests | Actions | Projects | Security | Insights

0 stars | 0 forks | 0 watching | Branches | Activity | Custom properties
Tags

Private repository

1 Branch | 0 Tags | Go to file t | Go to file | Code

howardginsburg fix to pass 'context' around so app insights picks up all log mess...

e5df217 · 5 months ago

.devcontainer	Initial commit of image resizi...	8 months ago
.vscode	refactor of facial recognition...	7 months ago
media	update to flow diagram	7 months ago
shared	fix to pass 'context' around s...	5 months ago
.funcignore	orchestration function and t...	7 months ago
.gitignore	update to gitignore	8 months ago
Readme.md	update to sample json to refl...	5 months ago
batchimagescaler.py	Added a blob trigger functio...	6 months ago
function_app.py	Added a blob trigger functio...	6 months ago
function_blob.py	fix to pass 'context' around s...	5 months ago
function_http.py	fix to pass 'context' around s...	5 months ago
host.json	Added a blob trigger functio...	6 months ago
image_processor.py	fix to pass 'context' around s...	5 months ago
local.settings.sample.j...	update to sample json to refl...	5 months ago

local_test_harness.http	Added a blob trigger functio...	6 months ago
requirements.txt	Added a blob trigger functio...	6 months ago

README

Image Ingest Pipeline

This Azure Functions project is a pipeline for reading images from a blob storage container, resizing them, and processing them through the Azure Face API to determine unique faces, and attempt to identify 'celebrity' faces.

Prerequisites

1. Azure Storage Account
   - You must create four containers in the storage account: ex: `images`, `resized`, `results`, `upload`.
2. Azure AI Services Account (Face API)
   - You must submit a request to enable usage of the full Face API including face similarity and celebrity recognition.
3. Azure Open AI
   - You must have an Azure OpenAI account and a GPT-4 Turbo or GPT4o deployment.

Getting Started

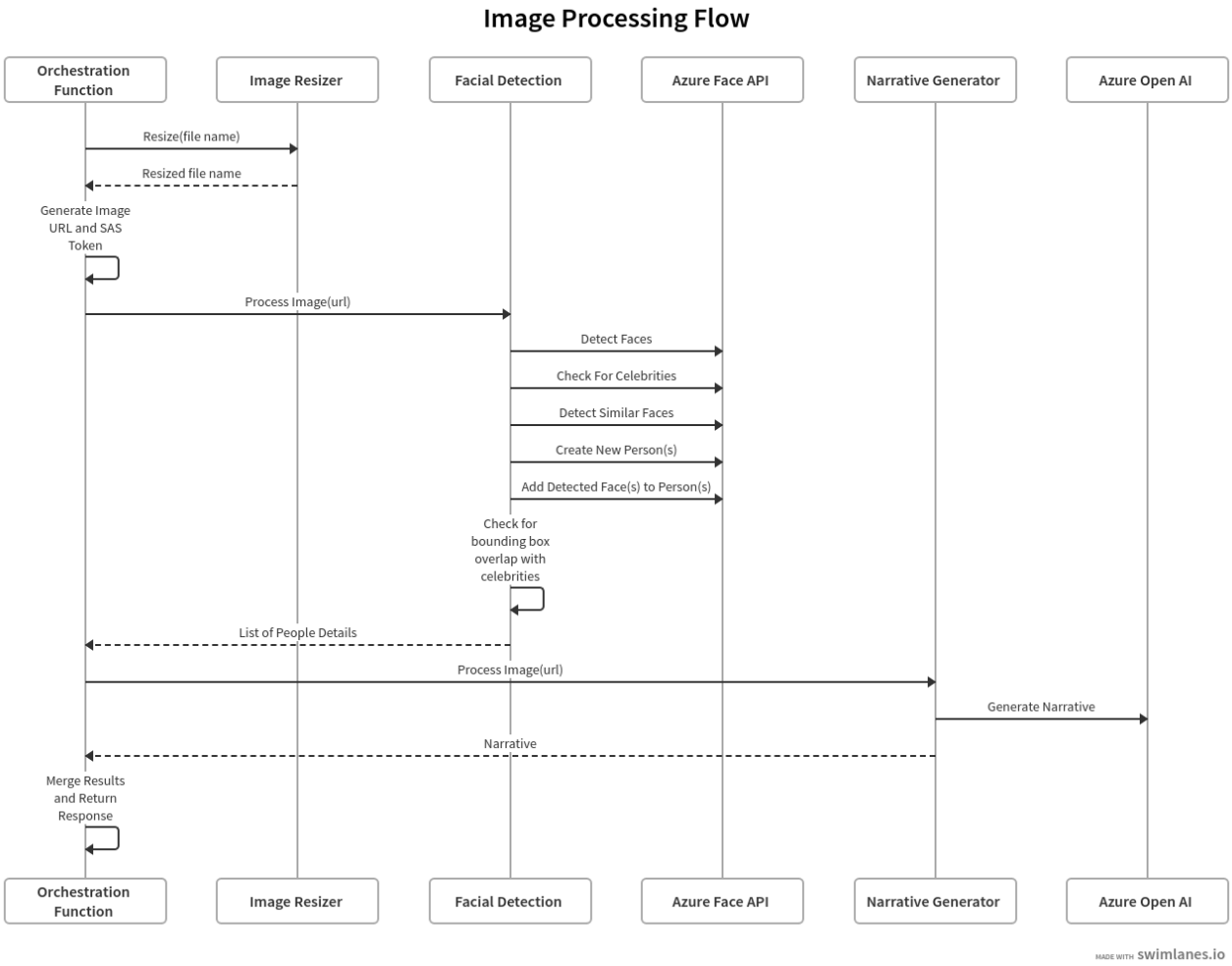
1. This repo is setup using devcontainers. You must have docker installed or run it as a Codespace in Github. If you plan to run it locally, after the project opens, make sure to select 'Yes' when prompted to reopen in container. This will ensure the correct environment is setup for the project, including the Azure Functions runtime environment.
2. Copy the `local.settings.json.example` file to `local.settings.json` make updates.
   - `AZURE_AI_SERVICE_ENDPOINT` is the endpoint for the Azure AI Service. For example: `https://<appname>.cognitiveservices.azure.com/`
   - `AZURE_AI_SERVICE_KEY` is the key for the Azure AI Service.
   - `AZURE_STORAGE_CONNECTION` is the connection string for the Azure Storage Account.

- `ORIGINAL_IMAGE_CONTAINER` is the name of the container where the original images are stored. For example: `images`
- `RESIZED_IMAGE_CONTAINER` is the name of the container where the resized images are stored. For example: `resized`
- `UPLOAD_IMAGE_CONTAINER` is the name of the container where the blob trigger function listens. For example: `upload`
- `ORCHESTRATOR_RESULT_CONTAINER` is optional. The name of the container where the face results are stored. For example: `results` If not set, then the json output of the function will not be stored.
- `ORCHESTRATOR_RESULT_CONNECTION` is optional. If you want to store the results in a separate storage account, then set this value to the connection string of the storage account. If `ORCHESTRATOR_RESULT_CONTAINER` is set and this value is not, then the `AZURE_STORAGE_CONNECTION` will be used.
- `AZURE_OPEN_AI_ENDPOINT` is the endpoint for the Azure OpenAI Service. For example: `https://<your openai name>.openai.azure.com/openai/deployments/<your gpt4 turbo/4o deployment>/chat/completions?api-version=2024-05-01-preview`
- `AZURE_OPEN_AI_KEY` is the key for the Azure OpenAI Service.

3. You can run the project locally by clicking the debug button in VSCode and selecting 'Attach to Python Functions'.

Architecture Overview

This project contains a single Azure Function endpoint that orchestrates several capabilities.



ImageScaler

Takes the name of an image file in the `images` container, resizes it, and uploads it to the `resized` container.

FaceRecognizer

Releases

No releases published

Packages

No packages published

Languages

- **Python** 99.8%
- **Dockerfile** 0.2%