digital-punchlist

Runtime Prep

If GNU make is installed on the system this is run on then `make run` will run the program. If make is not installed `python3 -m digital_punchlist` If not using make, Make sure the dependiencies are installed to run the program: `pip3 install -r requirements.txt`.

Environment File: Create file named .env in the root directory. Add the 3 required keys ("ACCESS_TOKEN", "EN_ACCESS_KEY", "EN_SECRET_KEY") for the Bazefield token and the two ensight keys respectively. Example format (../.env): ACCESS_TOKEN= EN_ACCESS_KEY= EN_SECRET_KEY=

Runtime modes

- Normal Engine Runtime: `python -m digital_punchlist "ENGINE"`
- Developer Normal Runtime: `python -m digital_punchlist "DEVELOPER" "[insert_name_of_config].json"`
- Developer Runtime to List Ensight Events @ sites/start/end in config: `python -m digital_punchlist "DEVELOPER_ENSIGHT" "[insert_name_of_config].json"`

Using json config for Developer Testing

Example Format for `[insert_name_of_config].json`:

```
{
    "listPluginsToTest": ["ALARMS-PXXX", "TREND-PYYY", ...],
    "listSitesToInclude": ["SITE1", "SITE2", ...],
    "excludeAssetsIfAny": ["SITE1-WTG050", ...],
    "startDateYYYYmmdd": "20220901",
    "endDateYYYYmmdd": "20221007",
    "csvOutputFilename": "output_filename"
}
```

- "listPluginsToTest": list of plugin script names as strings, engine will use this list to generate configuration for developer runtime without installing plugin in BF config tool
- "listSitesToInclude": list of site short names as strings, engine will use this list to generate configuration for developer runtime without installing plugin in BF config tool
- "excludeAssetsIfAny": list of Assets to IGNORE, leave empty list if none
- "startDateYYYYmmdd": flexible start date of developer analysis
- "endDateYYYYmmdd": flexible end date of developer analysis
- "csvOutputFilename" is a string that allows for custom naming of output CSV File

API Library Example

To get a list of assets at a site use `api.get_asset_names_at_site(site)`, returns a list of all assets at that site

Writing a plugin

Plugins are placed in the plugin folder and have the same name as the criteria code that it calculates (e.g. A plugin for P160 has a corresponding file `plugins/P160.py`, capitalization matters).

Plugin files need to contain a function `run` with the following signature:

```
 from digital_punchlist.bazefield_api import BazefieldAPI, OutputItem
from typing import List

def run(
    api: BazefieldAPI,
    site: str,
    turbines: List[str],
    start_times: List[datetime],
    end_times: List[datetime],
) -> List[OutputItem]:
```

The code in `run` will be will be executed everytime a criteria is calculated.

- `api` is an object that can be used to retrieve data from the Bazefield API. To get a list of methods and what they do run `help(BazefieldAPI)` or refer to the docs
- `site` is the site that the asset belongs to
- `turbines` A list of turbines at the site
- `start_times` is the start ranges that the criteria should be calculated for each turbnie (start_times[i] corresponds to turbines[i])
- `end_times` is the end range that the criteria should be calculated for each turbine (end_times[i] corresponds to turbines[i]

The run function is expected to calculate the criteria for all turbines on a site. Returns a list dictionary of the following form,

```
 [
    {
        "asset": "string"
        "priority": "string",
        "certainty": "string",
        "occurrence_start": "string",
        "occurrence_end": "string",
        "estimated_occurrence_loss": float,
        "estimated_aep_loss": float,
        "estimated_life_reduction_days": int,
        "status": "string",
    }
]
```

Each item in the list corresponds with one asset. The data that is returned will be uploaded to the BF database, as well as inserted into the csv generated at the end of runtime.

Writing an Ensight Specific Plugin

When writing a new Ensight plugin, user must identify hcCode of the issue from ensight, and once confirming that hcCode is not yet a created plugin, a new plugin can be made using that hcCode and alarmCategory details and the template below:

```
 from datetime import datetime, timedelta
from typing import List

from digital_punchlist.bazefield_api import BazefieldAPI, OutputItem

TIMESTAMP_FORMAT = '%Y-%m-%dT%H:%M:%S.0000000Z'
```

```
def run(
    api: BazefieldAPI,
    site: str,
    turbines: List[str],
    start_time: datetime,
    end_time: datetime,
) -> List[OutputItem]:
    ret: List[OutputItem] = []

    site_full_name = api.get_site_full_name_from_short_name(site)

    ensight_events: dict = api.get_ensight_events_for_wtg_list(
        turbines,
        site_full_name,
        (start_time[0]-timedelta(days=1)).strftime(TIMESTAMP_FORMAT),
        (end_time[0]-timedelta(days=0)).strftime(TIMESTAMP_FORMAT)
    )

    for asset in ensight_events:
        if (
            asset['hcCode'] == "[*INSERT_hcCode_OF_ISSUE_HERE*]"
        ):
            ret.append(
                {
                    "asset": asset['wtgName'],
                    "priority": "Next Visit",
                    "certainty": "Monitor",
                    "occurrence_start": start_time[0],
                    "occurrence_end": end_time[0],
                    "estimated_occurrence_loss": -1*asset['aepLoss']/52,
                    "estimated_aep_loss": -1*asset['aepLoss'],
                    "estimated_life_reduction_days": 0,
                    "status": "Office",
                    "evidence": [{
                        "timestamp": api.datetime_to_unix_time(start_time[0]),
                        "variable": "Ensight: Derates",
                        "value": -1*asset['aepLoss'],
                        "link":  api.get_ensight_link_from_site_and_type(
                            # Evidence link options are:
                            # "derates", "health", "controls", "sensors"
                            site, "derates"
                        ),
                    }],
                }
            )

    return ret
```

Priority Options

User also has options for changing "priority" to any of: ["Stop", "ASAP", "Next Visit", "Next PM", "Monitor"]

Status Options

User also has options for changing "status" of return object to any of: ["Site", "Office", "Auto-close"]