T-REX – Web APP Server

The Solar APP provides basic rest API for managing and executing pipeline over solar data associated to the NREL Web Site.

The App involves 4 main entities:

1. **Sites**
   Holding information about the site such as location, name and ID

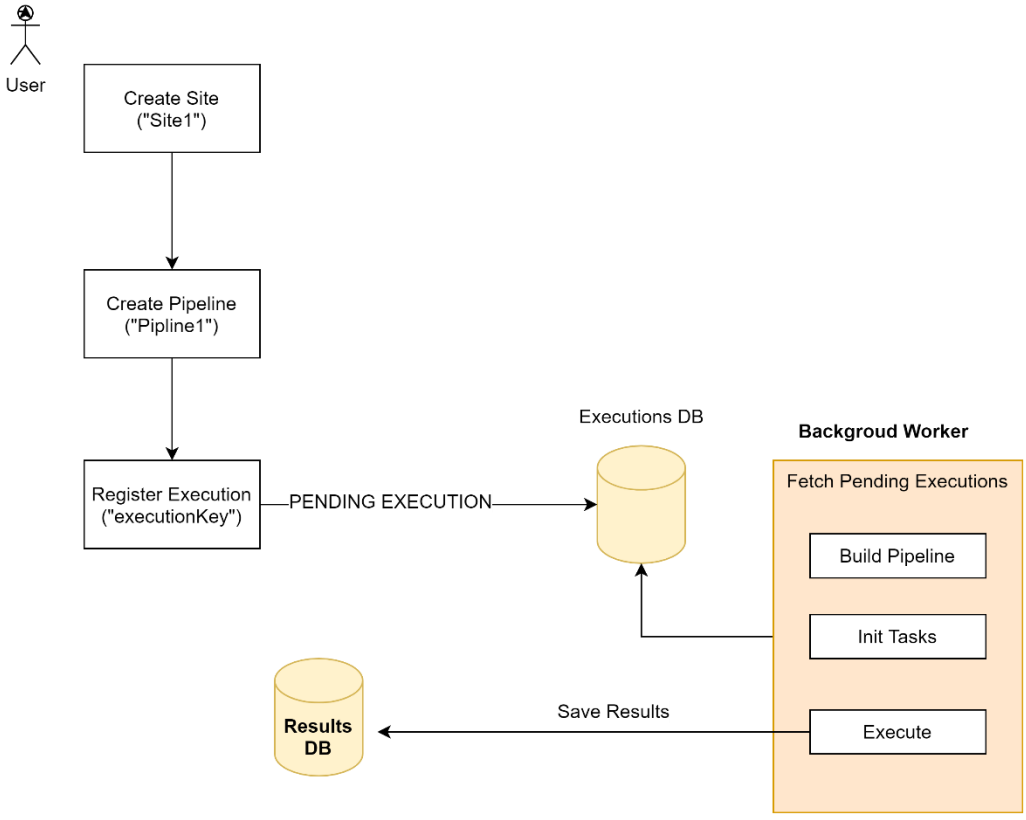
2. **Pipelines**
   Holding information about the tasks needs to be executed and the associated site.

3. **Execution**
   Represents a single execution of a site with a given pipeline.
   User can execute a site with unlimited times on a given pipeline, this as long as the previous pipeline execution is finished.

4. **Execution Result**
   Represents the result of running the pipeline, the result is saved by the **“Load”** task.
   For now it is supported of saving to either DB or File (CSV/Json)

Workflow

The following sequence describes general flow of a successful pipeline running:



Available Tasks for use in Pipeline

Here are the supported tasks implemented for this assignment:

Name	Type	Description
NrelSolarHttpTaskExtract	Extract	Fetch the Solar Data from NREL Web API
NrelSolarFlatTaskTransform	Transform	Transform Solar Data into flatten array of records of type “SolarMetric”
NrelSolarDbTaskLoad	Load	Save Results to DB Currently supporting the local NeDB
NrelSolarFileTaskLoad	Load	Saving Results to File Formats supported: CSV / JSON

Examples of REST API supported:

1. Adding new Site

REQUEST:

```
POST http://127.0.0.1:3000/sites/add
```

BODY

```
{
       "site": {
              "name": "MySite1",       // Site Name
              "id": 1,                 // Site ID
              "location": {
                     "lat": 40,
                     "lon":-120
              }
       }
}
```

RESPONSE:

```
{
    "id": 1,
    "status": "Success",
    "message": "Site Added Successfully"
}
```

2. Getting Site (Site ID = 1)

REQUEST:

```
GET http://127.0.0.1:3000/sites/1
```

RESPONSE

```
{
    "site": {
        "id": 1,
        "name": "MySite1",
        "location": {
            "lat": 40,
            "lon": -120
        },
        "pipeline": {
            "id": -1
        }
```

```
    },
    "status": "Success"
}
```

3. Getting All Sites

REQUEST:

```
GET http://127.0.0.1:3000/sites
```

RESPONSE

```
{
    "sites": [
        {
            "id": 2,
            "name": "MySite2",
            "location": {
                "lat": 40,
                "lon": -120
            },
            "pipeline": {
                "id": -1
            }
        },
        {
            "id": 1,
            "name": "MySite1",
            "location": {
                "lat": 40,
                "lon": -120
            },
            "pipeline": {
                "id": -1
            }
        }
    ],
    "status": "Success"
}
```

4. Add Pipeline

REQUEST:

```
POST http://127.0.0.1:3000/pipelines/add
{
      "pipeline": {
            "name": "MyPipeline",
            "id": 1,
            "tasks": {
                  "extract": {
                        "name": "NrelSolarHttpTaskExtract",
                        "input": {
                              "uri":
"https://developer.nrel.gov/api/solar/solar_resource",
                              "port": 443,
                              "version": 1,
                              "format": "json",
                              "uriParameters": {
                                    "api_key": "DEMO_KEY"
                              }
                        }
                  },
                  "transform": {
                        "name": "NrelSolarFlatTaskTransform",
                        "input": {}
```

```
                    },
                    "load": {
                          "name": "NrelSolarDbTaskLoad",
                          "input": {
                                "dbService": "NeDB",
                                "serviceArgs": {
                                      "dbName": "results"
                                }
                          }
                    }
              }
        }
}
```

If needed to output to file replace load task with the following:

```
                    "load": {
                          "name": "NrelSolarFileTaskLoad",
                          "input": {
                                "outputFile": "D:/result.csv",
                                "format": "csv"
                          }
                    }
```

- Set the location of the output file
- Set the desired format “csv” or “json”

RESPONSE:

```
{
    "id": 1,
    "status": "Success",
    "message": "Pipeline added Successfully"
}
```

5. Get Pipelines(s)

REQUEST:

GET http://127.0.0.1:3000/pipelines

RESPONSE:

Json Array contains all pipeline definitions

6. Get Pipeline

REQUEST:

GET http://127.0.0.1:3000/pipelines/1

RESPONSE:

Pipeline Definition associated with ID “1”

7. Execute Site with Pipeline

REQUEST:

PUT http://127.0.0.1:3000/sites/execute

```
{
```

```
      "siteId": 1,
      "pipelineId": 1
}
```

- Will run site with id=1 with pipeline id=1

RESPONSE:

The response contains information about the execution, such as the execution key site, pipeline and times.
The initial status is PENDING.


```
{
    "executionInfo": {
        "key": "b8e77e6c-f3f7-4d55-92f9-ccb2a3e4eb7d",
        "siteId": 1,
        "pipelineId": 1,
        "status": "PENDING",
        "createTime": "2021-05-25T00:53:58.712+03:00",
        "updateTime": "2021-05-25T00:53:58.712+03:00",
        "createTimeUtc": 1621893238,
        "updateTimeUtc": 1621893238
    },
    "status": "Success",
    "message": "Execution added Successfully"
}
```


8. Get Last Execution Status

- Add the "Execution Key" to the URL query params

REQUEST:

GET http://127.0.0.1:3000/executions?key=b8e77e6c-f3f7-4d55-92f9-ccb2a3e4eb7d

RESPONSE:


```
{
    "executions": [
        {
            "key": "b8e77e6c-f3f7-4d55-92f9-ccb2a3e4eb7d",
            "siteId": 1,
            "pipelineId": 1,
            "status": "COMPLETED",
            "createTime": "2021-05-25T00:53:58.712+03:00",
            "updateTime": "2021-05-25T00:54:00.019+03:00",
            "createTimeUtc": 1621893238,
            "updateTimeUtc": 1621893240
        }
    ],
    "status": "Success"
}
```


9. Get All Executions

REQUEST

GET http://127.0.0.1:3000/executions

GET http://127.0.0.1:3000/executions?status=COMPLETED

RESPONSE:

```
{
    "executions": [
        {
            "key": "21329173-aa39-4910-bf0c-6c9d457dbd18",
            "siteId": 1,
            "pipelineId": 1,
            "status": "COMPLETED",
            "createTime": "2021-05-24T23:31:08.785+03:00",
            "updateTime": "2021-05-24T23:31:34.320+03:00",
            "createTimeUtc": 1621888268,
            "updateTimeUtc": 1621888294
        },
        {
            "key": "f7cd570c-7da9-4b69-9b06-7bcee186e29e",
            "siteId": 1,
            "pipelineId": 3,
            "status": "COMPLETED",
            "createTime": "2021-05-24T23:22:25.316+03:00",
            "updateTime": "2021-05-24T23:23:47.365+03:00",
            "createTimeUtc": 1621887745,
            "updateTimeUtc": 1621887827
        }
    ],
    "status": "Success"
}
```


Available Query Parameters for Filtering:

status	COMPLETED PENDING RUNNING FAULTED CANCELLED
key	Specific Execution Key
siteId	Executions related to specific Site ID

10. Get Execution Result

```
// Getting all executions results
GET http://127.0.0.1:3000/executions/results

// Getting specific execution results matches execution key
GET http://127.0.0.1:3000/executions/results/21329173-aa39-4910-bf0c-
6c9d457dbd18

In case of Load task which writes to File, the result is a metadata record
describes the execution info, for example:
```

GET http://127.0.0.1:3000/executions/results/438b27c4-b0f6-4282-bf5c-75f713668723


```
{
    "status": "Success",
    "results": [
        {
            "executionKey": "438b27c4-b0f6-4282-bf5c-75f713668723",
            "file": "D:/nrel-result.csv",
            "format": "csv",
            "status": "COMPLETED",
            "createTime": "2021-05-25T01:04:11.612+03:00",
            "updateTime": "2021-05-25T01:04:11.612+03:00",
            "createTimeUtc": 1621893851,
```

```
            "updateTimeUtc": 1621893851
        }
    ]
}
```


Example of Solar Metric Record Saved to DB

GET http://127.0.0.1:3000/executions/results/affbef68-8a27-4300-9f06-7b70b5711393

RESPONSE


```
{
    "status": "Success",
    "results": [
        {
            "siteName": "MySite1",
            "siteId": 1,
            "pipelineName": "MyPipeline",
            "pipelineId": 1,
            "executionKey": "affbef68-8a27-4300-9f06-7b70b5711393",
            "source": "Perez-SUNY/NREL, 2012",
            "version": "1.0.0",
            "type": "avg_lat_tilt",
            "period": "February",
            "value": 4.89,
            "createTime": "2021-05-25T01:08:22.943+03:00",
            "updateTime": "2021-05-25T01:08:22.943+03:00",
            "createTimeUtc": 1621894102,
            "updateTimeUtc": 1621894102
        },
        {
            "siteName": "MySite1",
            "siteId": 1,
            "pipelineName": "MyPipeline",
            "pipelineId": 1,
            "executionKey": "affbef68-8a27-4300-9f06-7b70b5711393",
            "source": "Perez-SUNY/NREL, 2012",
            "version": "1.0.0",
            "type": "avg_dni",
            "period": "June",
            "value": 9.3,
            "createTime": "2021-05-25T01:08:22.943+03:00",
            "updateTime": "2021-05-25T01:08:22.943+03:00",
            "createTimeUtc": 1621894102,
            "updateTimeUtc": 1621894102
        }
        ...
    ]
}
```

Running Notes:

1. It is required to have NodeJS installed with NPM on a computer
2. Once extracted all the files, enter the “server” folder and run:

```
npm run rebuild
```

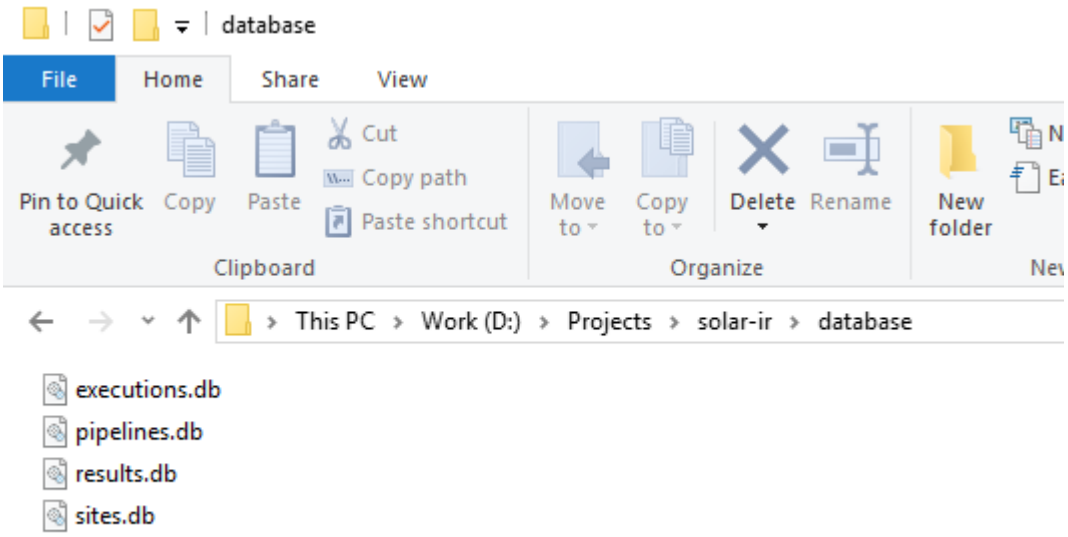
3. Once completed, run the server by running:

```
npm run start
```

4. The server listens on port 3000 (default):

```
App listening on port 3000
25-05-2021 01:12:45.358|INFO| Monitoring Enabled, polling 10 seconds !
25-05-2021 01:12:46.599|INFO| PipelineExecutorWorker, polling executions
...
Found Total 0 Records
25-05-2021 01:12:46.631|INFO| PipelineExecutorWorker, No executions found,
will try again in 10 seconds ...
25-05-2021 01:12:46.632|INFO| PipelineExecutorWorker, checking expired
executions ...
Found Total 0 Records
25-05-2021 01:12:46.635|INFO| PipelineExecutorWorker, No expired
executions found !
```

5. DB Files are created locally under “database”
   If needed to clear you can delete them and they will be created empty.



6. Logs are under “logs” folder

Content Provided:

1. Application Source Files
2. POSTMAN Files as examples to be used in Chrome (can be imported and use)
3. CSV Result Example
4. JSON Result Example
5. This Briefly Document

Sorry for the delay … had busy time with kids.

Enjoy
Raz