

# Import Prices

Example of how automate daily import of Prices for Client Portfolios

```
com/finmars/client-daily-schedule/import-prices/workflow.json
```

```
{
  "workflow":
  {
    "user_code":"com.finmars.client-daily-schedule:import-prices",
    "is_manager": true,
    "tasks": ["com.finmars.client-daily-schedule:import-prices.task"],
    "periodic": {
      "crontab": "30 2 * * *"
    }
  }
}
```

```
com/finmars/client-daily-schedule/import-prices/tasks.py
```

This simple code basically does

- get prices
- transform prices
- import prices

```
import jwt
import copy
import json
import requests
from datetime import datetime, timezone, timedelta
from requests.auth import HTTPBasicAuth
import traceback
import time

from workflow.api import task
```

```
from workflow.finmars import storage, utils, vault, request_api, create_logger
```

```
workflow_user_code="com.finmars.client-daily-schedule:import-prices"
```

```
@task(name="%s.task" %workflow_user_code, bind=True)
```

```
def main(self, *args, **kwargs):
```

```
    # Expected payload
```

```
    # payload = {
```

```
    #   "date": "2024-09-23"
```

```
    # }
```

```
    # TODO we need connection between Portfolio and Client and Vault Keypath, right now hardcoded
```

```
    # portfolios = get_portfolios()
```

```
    # Get today's date
```

```
    today = datetime.now()
```

```
    # Calculate yesterday's date
```

```
    yesterday = today - timedelta(days=1)
```

```
    payload = kwargs.get("payload", {})
```

```
    target_date = payload.get('date', str(yesterday.date()))
```

```
    instruments = get_instruments()
```

```
    portfolios = [
```

```
        {
```

```
            "secret": "finmars/exante",
```

```
            "user_code": "ABCD123.001"
```

```
        },
```

```
    ]
```

```
    self.log(f"Target Date {target_date}")
```

```

workflow_get_positions = self.workflow.run_new_workflow(
    "com.finmars.exante-broker:get-prices",
    payload={
        "secret": portfolios[0]['secret'],
        "date_from": target_date,
        "date_to": target_date,
        "instruments": instruments
    }
)

poll_workflow_status(workflow_get_positions['id'])

self.log(f"Prices Received from Exante")

workflow_transform_positions = self.workflow.run_new_workflow(
    "com.finmars.exante-data-transformer:transform-prices",
    payload={
        "date_from": target_date,
        "date_to": target_date,
        "instruments": instruments
    }
)

poll_workflow_status(workflow_transform_positions['id'])

self.log(f"Prices Transformed to Standard Configuration")

for instrument in instruments:
    workflow_import_positions = self.workflow.run_new_workflow(
        "com.finmars.standard-workflow:simple-autoimport",
        payload={
            "actions": [
                {
                    "file_path":
f"/data/general/prices/exante_{convert_to_ascii(instrument)}_{target_date}_{target_date}.json",
                    "import_type": "data",
                    "schema": "com.finmars.standard-import-from-file:instruments.pricehistory:price"
                }
            ]
        }
    )

```

```
)
```

```
# poll_workflow_status(workflow_transform_positions['id'])
```

```
self.log(f"Prices Imported")
```

```
# should be moved to Finmars Workflow Library
```

```
def poll_workflow_status(workflow_id, max_retries=100, wait_time=5):
```

```
    url = f'/workflow/api/workflow/{workflow_id}/' # Replace with your actual API endpoint
```

```
    for attempt in range(max_retries):
```

```
        data = request_api(url)
```

```
        if data:
```

```
            status = data.get('status')
```

```
            print(f'Attempt {attempt + 1}: Workflow status is {status}')
```

```
            if status in ['success', 'error']:
```

```
                return status # Return the status when it's success or error
```

```
        else:
```

```
            print(f'Error fetching status: {response.status_code}')
```

```
        time.sleep(wait_time) # Wait before the next attempt
```

```
    print('Max retries reached. Workflow status not successful.')
```

```
    return None # Indicate that the status was not found
```

```
def get_instruments():
```

```
    data = request_api('/api/v1/instruments/instrument/light/')
```

```
    instruments = []
```

```
    for item in data['results']:
```

```
        if item['short_name'] != '-':
```

```
            instruments.append(item['short_name'])
```

```
    return instruments
```

```
def convert_to_ascii(input_string):

    input_string = input_string.lower()

    # Convert spaces and dots to underscores
    modified_string = input_string.replace(' ', '_').replace('.', '_')

    # Convert the string to ASCII, ignoring non-ASCII characters
    ascii_string = modified_string.encode('ascii', 'ignore').decode()

    return ascii_string
```

In the end you able to see you workflow Scheduled in Finmars Workflow

com.finmars.itech-daily-schedule:import-prices		30 2 *** 🕒	Manager	▶
--	--	------------	---------	---

Revision #6

Created 30 September 2024 23:25:12 by Sergei Zhitenev

Updated 30 September 2024 23:46:24 by Sergei Zhitenev