BIGDATA TEAM

HW #08: Asset Web Service

1. Описание задания

В данном ДЗ нужно написать финансово-аналитический Web Service, который позволит мониторить изменение курса валют и их влияние на инвестиционные продукты. Для решения задания вам пригодятся умения:

- парсить HTML (например с помощью bs4, lxml и/или XPath);
- использовать паттерны проектирования (например Composite);
- тестировать и поднимать Web-сервисы на Python с помощью Flask и pytest.

1.1. Получение актуальных данных с сайта ЦБР

В рамках предыдущих учебных модулей вы уже могли познакомиться с консольным приложением asset.py. Консольное приложение (и библиотека) предоставляет возможность рассчитать инвестиционную привлекательность актива за интересующий период в будущем:

- https://github.com/big-data-team/python-course/asset.py

В рамках этого задания предлагается расширить функциональность библиотеки по работе с активами в разной валюте, учитывающую актуальные значения курсов валют на сайте Центрального Банка Российской Федерации (cbr.ru).

Ваш код должен содержать функции по парсингу двух HTML-страниц сайта cbr.ru. Каждая функция парсинга будет принимать один аргумент - строку с всем содержимым HTML-документа. Результатом парсинга будет является словарь[1]:

- {“char_code”: rate}
- char_code - буквенный код
- rate - курс валют в обмене на 1 у.е.

Добавьте следующие функции (и конечно же соответствующие тесты) со следующей сигнатурой:

```
def parse_cbr_currency_base_daily(html_data: str) -> Dict[str,float]:
    # content example: https://www.cbr.ru/eng/currency_base/daily/
    # dump example: github../cbr_currency_base_daily.html
```

[1] Точность сравнения дробных чисел - 10e-8

```python
def parse_cbr_key_indicators(html_data: str) -> Dict[str,float]:
    # content example: https://www.cbr.ru/eng/key-indicators/
    # dump example: github../cbr_key_indicators.html
```

Обратите внимание, что в версии страниц на английском языке.

Сам сервис (Flask-приложение) должен предоставлять следующие route'ы:

endpoint: /cbr/daily
формат ответа: JSON
описание: сделать запрос на страницу "daily" и получить значения курсов валют в формате {"char_code": rate}.

endpoint: /cbr/key_indicators
формат ответа: JSON
описание: сделать запрос на страницу "key-indicators" и получить значения для курса валют USD, EUR и драгоценных металлов в формате {"char_code": rate}.

замечание: фильтрацию по длине char_code делать нельзя, в тестах будут встречаться и длинные имена, можно использовать только структуру HTML и указание классов тегов для парсинга нужных значений.

Обработка исключительных ситуаций:

- В случае обращения по несуществующему route должен возвращаться **код 404** с текстом "This route is not found";
- В случае недоступности cbr.ru (ожидаемый exception - requests.**exceptions.ConnectionError**), необходимо возвращать ошибку 503. Для этой ошибки должен быть зарегистрирован обработчик, который будет возвращать сообщение "CBR service is unavailable". Для тестирования этого поведения необходимо будет запатчить обращения "requests.get".

1.2. Сервис по работе с активами

Имея актуальные данные по курсам валют предоставьте возможность работать с портфелем активов в формате Web-сервиса. В дополнение к уже имеющимся route сервис должен предоставить следующую функциональность:

endpoint: /api/asset/add/char_code/name/capital/interest
формат ответа: text

описание: добавить актив в валюте “char_code” с именем “name”, размером капитала capital и оценочной инвестиционной годовой доходностью interest (в процентах, записанных дробным числом; то есть в качестве interest можно указать число 0.5, что будет означать 50%). Запрос должен возвращать код возврата 200 и сообщение "Asset '{name}' was successfully added". В случае попытки добавления актива с именем (name), который уже существует в базе, система должна выдавать код возврата 403.

замечание: обратите внимание на раздел FAQ, где описаны особенности указания численных converter’ов Flask

рекомендации: для хранения всех активов вам пригодится шаблон проектирования Composite и глобальное хранилище активов (например, сохраните в переменную app.bank[2]).

endpoint: /api/asset/list
формат ответа: JSON
описание: вернуть список всех доступных активов, каждый актив представить списком [“char_code”, “name”, capital: float, interest: float] Сортировка списков по умолчанию: сначала по возрастанию “char_code”, затем по остальным полям.

endpoint: /api/asset/get?name=name_1&name=name_2
формат ответа: JSON
описание: вернуть список всех перечисленных активов, каждый актив представить списком [“char_code”, “name”, capital, interest]. Сортировка списков по умолчанию: сначала по возрастанию “char_code”, затем по остальным полям.

endpoint: /api/asset/calculate_revenue?period=period_1&period=period_2
формат ответа: JSON
описание: рассчитать оценочную инвестиционную доходность в рублях за указанные периоды времени. Вернуть словарь {“period”: revenue}, где для валют USD, EUR и драгоценных металлов делать обращения на страницу “key-indicators” (в противном случае реализация будет посчитана неверной), а остальные со страницы “daily”. (точность сравнения дробных чисел 10e-8.) Также учитывайте, что может быть и рублевый актив.

endpoint: /api/asset/cleanup
формат ответа: text
описание: очистить список активов. Запрос должен возвращать код возврата - 200 и сообщение “there are no more assets”.

[2] Вопросы thread-safe и корректной работы в проде через multiprocessing - правильные, но выходят за рамки текущего учебного модуля.

1.3. Требования к реализации

Flask приложение должно называться app и создаваться с помощью конструкции:

```
app = Flask(__name__)
```

Тестирующая система использует конструкцию:

```
from task_<Surname>_<Name>_asset_web_service import app
```

Проверки и возможное поведение системы:

- все route'ы имеют тип HTTP-запроса - GET;
- все обращения в интернет должны производиться с помощью вызова "requests.get" (именно такие запросы будут Mock'аться в тестах);
- гарантируется, что надежные ответы дают обращения только к следующим публичным сущностям класса requests.models.Response: "status_code", "encoding", "text", "content" (последние три только в случае status_code == 200). Другие могут выдавать правдоподобные ответы, но полагаться на них не стоит;
- будет проверяться только исключительное поведение указанное в условии. Тем не менее, выявлять и обрабатывать все случаи неожиданного поведения, неправильного ввода и т.п. - полезное упражнение;
- учитывайте, что как и реальный пользователь тестирующая система может за одну сессию переходить по нескольким адресам вашего веб-приложения;
- в HTML-документах, которые будут использованы при тестировании (как парсинга, так и работы веб-сервиса) могут быть удалены некоторые непопулярные указанные на сайте и добавлены выдуманные валюты для проверки того, что парсер и код веб-сервиса не привязан к конкретному набору валют;
- ваша программа не должна в ходе работы печатать что-либо в stdout. Это может привести к неинформативным ошибкам в отчете по тестированию.

2. Рекомендации

Рекомендации по разработке:

- следите за качеством кода и проверяйте "глупые" ошибки с помощью pylint, следите за поддерживаемостью и читаемостью кода;
- держите уровень покрытия кода тестами на уровне 80+%, следуйте TDD (сначала тесты, потом реализация);
- отделяйте фазу рефакторинга от фазы добавления новой функциональности.

    - фиксируем функциональность, все тесты зеленые;
    - проводим рефакторинг;
    - по окончании фазы рефакторинга снова все тесты зеленые;
- следите за скоростью выполнения unit-test'ов, несколько секунд - это хорошо, в противном случае нужно уменьшать размер тестируемых датасетов или разделять тесты на фазы (см. обсуждение про mark.slow);

3. Критерии оценивания

Балл за задачу складывается из:

- **30%** - правильная реализация парсинга
- **40%** - правильная реализация API Web-сервиса
- **20%** - качество покрытия юнит-тестами
    - оценка качества проводится автоматически вызовом pytest:
        - `PYTHONPATH=. pytest -v --cov=task_*_asset_web_service test_*_asset_web_service.py`
        - уровень покрытия тестами должен быть выше 80%
        - проверяем код Python версии 3.7 с помощью `pytest==6.0.1`
        - точная формула: 20% x min([test_coverage / 0.8], 1.0)
- **10%** - поддерживаемость и читаемость кода
    - в общем случае см. Clean Code и Google Python Style Guide
    - оценка качества будет проводиться автоматическим вызовом pylint:
        - `pylint task_*.py`
        - качество кода должно оцениваться выше 8.0 / 10.0
        - проверяем код Python версии 3.7 с помощью `pylint==2.5.3`
        - точная формула: 10% x min([lint_quality / 8.0], 1.0)

Discounts (скидки и другие акции):

- **100%** за плагиат в решениях (всем участникам процесса)
- **100%** за посылку решения после hard deadline
- **30%** за посылку решения в после soft deadline и до hard deadline
- **5%** за каждую посылку после 2й посылки в день (каждый день можно делать до 2х посылок без штрафа)

лучший балл с 1-й попытки: 100%
лучший балл со 2-й попытки: 100%
лучший балл с 3-й попытки: 95%
лучший балл с 4-й попытки: 90%

4. Инструкция по отправке задания

Оформление задания:

- Код задания (Short name): **HW08:Asset Web Service**
- Выполненное ДЗ запакуйте в архив **PY-MADE-2021-Q4**_<Surname>_<Name>_HW#**.zip**, пример -- **PY-MADE-2021-Q4**_Dral_Alexey_**HW08**.zip. (Проверяйте отсутствие пробелов и

невидимых символов после копирования имени отсюда.[3]) Если ваше решение лежит в папке my_solution_folder, то для создания архива hw.zip на Linux и Mac OS выполните команду[4]:
    - `zip -r hw.zip my_solution_folder/*`
- На Windows 7/8/10: необходимо выделить все содержимое директории my_solution_folder/ нажать правую кнопку мыши на одном из выделенных объектов, выбрать в открывшемся меню "Отправить >", затем "Сжатая ZIP-папка". Теперь можно переименовать архив.
- Решение задания должно содержаться в одной папке.
- Перед проверкой убедитесь, что дерево вашего архива выглядит так:
    - **`| PY-MADE-2021-Q4_<Surname>_<Name>_HW08.zip`**
    - **`| ---- task_<Surname>_<Name>_asset_web_service.py`**
    - **`| ---- test_<Surname>_<Name>_asset_web_service.py`**
    - **`| ---- *.{html,txt}`**[5]
    - При несовпадении дерева вашего архива с представленным деревом, ваше решение не будет возможным автоматически проверить, а значит, и оценить его.
- Для того, чтобы сдать задание, необходимо:
    - Зарегистрироваться и залогиниться в сервисе Everest
    - Перейти на страницу приложения: MADE Python Grader
    - Выбрать вкладку Submit Job (если отображается иная).
    - Выбрать в качестве “Task” значение: **HW08:Asset Web Service**[6]
    - Загрузить в качестве “Task solution” файл с решением
    - В качестве Access Token указать тот, который был выслан по почте
- **Перед отправкой задания**, оставьте, пожалуйста, отзыв о домашнем задании по ссылке: https://rebrand.ly/pymade2021q4_feedback_hw. Это позволит нам скорректировать учебную нагрузку по следующим заданиям (в зависимости от того, сколько часов уходит на решение ДЗ), а также ответить на интересующие вопросы.

Внимание: если до дедлайна остается меньше суток, и вы знаете (сами проверили или коллеги сообщили), что сдача решений сломана, обязательно сдайте свое решение, прислав нам ссылку на выполненное задание (Job) на почту с темой письма “Short name. ФИО.”. Например: “**HW08:Asset Web Service**. Иванов Иван Иванович.” Таким образом, мы сможем увидеть какое решение у вас было до дедлайна и сможем его оценить. Пример ссылки:

- https://everest.distcomp.org/jobs/67893456230000abc0123def

---

[3] Онлайн инструмент для проверки: https://www.soscisurvey.de/tools/view-chars.php
[4] Флаг -r значит, что будет совершен рекурсивный обход по структуре директории
[5] Архив с тестовыми данными должен занимать **менее 200 КБ** пространства на жестком диске
[6] Сервисный ID: python.asset_web_service

Любые вопросы / комментарии / предложения пишите согласно предложениям на портале.

Всем удачи!

5. FAQ (часто задаваемые вопросы)

"You are not allowed to run this application", что делать?
Если Вы видите надпись "You are not allowed to run this application" во вкладке Submit Job в Everest, то на данный момент сдача закрыта (нет доступных для сдачи домашних заданий, по техническим причинам или другое). Попробуйте, пожалуйста, еще раз через некоторое время. Если Вы еще ни разу не сдавали, у коллег сдача работает, но Вы видите такое сообщение, сообщите нам об этом.

Grader показывает 0 или < 0, а отчет (Grading report) не помогает решить проблему
Ситуации: • система оценивания показывает оценку (Grade) < 0, а отчет (Grading report) не помогает решить проблему. Пример: в случае неправильно указанного access token система вернет -401 и информацию о том, что его нужно поправить; • система показывает 0 и в отчете (Grading report) не указано, какие тесты не пройдены. Пример: вы отправили невалидный архив (rar вместо zip), не приложили нужные файлы (или наоборот приложили лишние - временные файлы от Mac OS и т.п.), рекомендуется проверить содержимое архива в консоли: `unzip -l your_solution.zip` Если Вы столкнулись с какой-то из них присылайте ссылку на выполненное задание (Job) в чат курса. Пример ссылки: https://everest.distcomp.org/jobs/67893456230000abc0123def

Описание теста test_can_parse_cbr_daily_output

Вы можете столкнуться с выводом:

test_can_parse_cbr_daily_output[task::python.asset_web_service]:
excinfo: AssertionError('Expected char_code AMD was not found in the output or its rate is incorrect. Make sure to use unit column properly')

Обратите внимание, что курс нужно брать за 1 условную единицу, а на странице cbr он может быть представлен за 100 или 1000 условных единиц:

Центральный банк Российской Федерации установил с 07.10.2021 следующие курсы иностранных валют к рублю Российской Федерации без обязательств Банка России покупать или продавать указанные валюты по данному курсу

Цифр. код	Букв. код	Единиц	Валюта	Курс
036	AUD	1	Австралийский доллар	52,5031
944	AZN	1	Азербайджанский манат	42,7123
051	AMD	100	Армянских драмов	14,8462
933	BYN	1	Белорусский рубль	29,0622
975	BGN	1	Болгарский лев	42,8839
986	BRL	1	Бразильский реал	13,2496
348	HUF	100	Венгерских форинтов	23,2945
410	KRW	1000	Вон Республики Корея	60,7113

Описание теста test_can_parse_cbr_key_indicators_output

Вы можете столкнуться с выводом:

- test_can_parse_cbr_key_indicators_output[task::python.asset_web_service]:
excinfo: 'AssertionError(''IndexError: list index out of range'')'

Обратите внимание, что страница на сайте и в дампе могут отличаться (нужно уметь парсить обе, проверяться будет на дампе). Одна из разниц приводящая к такой проблеме, что курс по драг. металлам приводится за одну дату, без истории.

Описание теста test_can_add_and_list_assets

Вы можете столкнуться с выводом:

- test_can_add_and_list_assets[task::python.asset_web_service]:

excinfo: AssertionError('assert False')

Для того, чтобы отвечать в формате json нужно задать правильный header (Content Type: application/json). Возвращать json.dumps из функции - это не тоже самое, что возвращать jsonify. Flask jsonify задает нужные параметры HTTP запроса

Проверить Flask Response объект можно с помощью вызова response.is_json. В одном случае будет True (jsonify), а в другом - False (что неправильно)

Численные конвертеры во Flask

Если в route прописать
<float:capital>
то при следующих вызовах возникнут проблемы:

```
(base) ➜  ~ curl -I http://127.0.0.1:5000/api/asset/add/USD/hi/1000/0.1
HTTP/1.0 404 NOT FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 23
Server: Werkzeug/1.0.1 Python/3.7.11
Date: Wed, 06 Oct 2021 15:58:55 GMT

(base) ➜  ~ curl -I http://127.0.0.1:5000/api/asset/add/USD/hi/1000.0/0.1
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 33
Server: Werkzeug/1.0.1 Python/3.7.11
Date: Wed, 06 Oct 2021 15:59:03 GMT
```

Grader будет содержать ошибки типа:
excinfo: AssertionError('assert 200 == 404')

Объяснение такого поведения можно найти в issue проекта Flask:

- https://github.com/pallets/flask/issues/315

Там вы увидите пояснение и пример еще одной проблемы (отрицательные float тоже не парсятся по умолчанию), поскольку регулярка в werkzeug на float следующая:
» \d+.\d+

это поправили в релизе 0.15 werkzeug с возможностью добавлять флаги в route:

https://github.com/pallets/werkzeug/issues/729

например (и int тоже поправили): <int(signed=True):count>

но вот конвертацию в int исправить не хотят (см. report 2019го года):
https://github.com/pallets/werkzeug/issues/1645

и рекомендуют писать просто 2 route:
@app.route('/venues/<int:lat>/<int:lon>/<int:rad>/')
@app.route('/venues/<float:lat>/<float:lon>/<float:rad>/')

Это работает, но на наш субъективный взгляд - это плохая практика. Поясним:

- если у вас два float параметра, то получите 4 возможных сочетания (остальные параметры просто дублируем во всех строчках), в случае 3х float-параметров будет вообще 8 строчек.

Наша рекомендация - кастовать к float напрямую в коде, будет один route (параметр по умолчанию - str) и максимально простой и работающий код.

Как правильно настроить окружение, чтобы оно совпадало с тестовым окружением?
1. Если еще не установлено, то установите conda https://docs.conda.io/projects/conda/en/latest/user-guide/install/ 2. Настройте окружение для разработки на основе README.md курса https://github.com/big-data-team/python-course 3. Скачайте необходимые датасеты для выполнения задания https://github.com/big-data-team/python-course#study-datasets

6. Дополнительные задания (не на оценку)

Вклад в государственный бюджет

Предположим, что у вас есть собственный актив, небольшое предприятие, где команда создает продукты интеллектуальной собственности (электронные произведения искусства). Такой актив работает следующим образом:

- capital: вложение в зарплату сотрудников, производящих продукты (исключаем все расходы на операционную деятельность, менеджеров, бухгалтерию и т.п.)

- продажа всех продуктов спустя год (считаем, что у вас есть постоянный покупатель, который покупает произведения искусства раз в год в свою коллекцию под Новый Год).
- interest: наценка на продукты, добавочная стоимость

Если инвестиции в актив идут несколько лет подряд, то все вырученные средства в один год вкладываются целиком на следующий.

Корреспондент спрашивает Рокфеллера как он разбогател.
- Когда я был маленький у меня было 2 цента. Я мог пойти в кино, но я купил грязное яблоко, помыл его и продал за 4 цента. На эти деньги я мог купить гамбургер, но я купил 2 грязных яблока, помыл их и продал за 8 центов. На эти деньги я мог пойти в ресторан, но я купил 4 грязных яблока, помыл их и продал за 16 центов, а потом умер мой папа и оставил мне 100 млн долларов.

Произведем оценочные вычисления по вкладу такого актива в бюджет страны различными отчислениями (рассчитываются от зарплаты сотрудника):

- Пенсионный Фонд России (ПФР) - 22%;
- Фонд Социального Страхования (ФСС) - 2.9%;
- Федеральный фонд обязательного медицинского страхования (ФФОМС) - 5.1%;
- на травматизм - от 0,2 до 8,5 % (для простоты будем считать 0.2%);
- ИТОГО - 30.2% страховых взносов.

Если учесть налог на доходы физических лиц (НДФЛ) в размере 13%, то в бюджет государства в совокупности будет уплачено от размера капитала:

- (1 - 1 / 1.302 * (1.0 - 0.13)) ≈ 0.332 (или 33.2%)[7]

Для простоты вычислений, налоги компании и НДС учитывать не будем.

Добавьте следующий route:

endpoint: /api/asset/calculte_country_tax?period=p1&period=p2
формат ответа: JSON
описание: вернуть словарь {“period”: [revenue, country_tax]}, где за каждый период указаны ваш доход и размер уплаченных налогов в бюджет страны в рублях.

[7] Если считать от чистой зарплаты (на руки), то сверху к этой зарплате будет добавлено (1 / 0.87 * 1.302 - 1) ≈ 49.7% для уплаты в государственные фонды