Ручное обновление курсов.

№ урока:	12	**Курс:**	Python Practice

Средства обучения:	Интерпретатор Python, virtualenv, текстовый редактор

Обзор, цель и назначение урока

Цель урока: расширить функционал созданного web приложения, добавить ручное обновление курсов. Познакомить студентов с основами работы с POST запросами во Flask, механизмом декораторов в Python. Выстроить простейшую систему защиты в приложении Golden-Eye.

Изучив материал данного занятия, учащийся:

- Научится обрабатывать POST запросы и получать содержимое POST запроса во Flask приложениях
- Узнает о дополнительных атрибутах запросов - HTTP метод запроса, заголовки, ip-адрес клиента и других.
- Узнает, что такое декораторы в Python и научится их создавать и использовать
- Закрепит навыки работы с ORM библиотекой peewee

Содержание урока

1. Краткое резюме, что мы уже имеем к данному момент и обзор дальнейших действий
2. Обзор HTTP методов запроса — GET, POST. Как указать ожидаемый метод во Flask
3. Работа с POST запросом во Flask приложении, получение данных запроса.
4. Ошибка Bad Request при доступе к данным запроса
5. Добавление функционала с ручным обновлением, проверка в браузере
6. Работа с дополнительными атрибутами запросов — метод, заголовки, ip адрес клиента, реферер, данные запроса.
7. Необходимость защитить отдельные url от несанкционированного доступа, простейший вариант реализации защиты
8. Декораторы в Python. Реализации системы защиты с помощью декораторов
9. Перенос просмотра логов в раздел "защищенных" url
10. Проверка внесенных изменений в браузере

Резюме

- Сегодня нам предстоит добавить новый функционал в приложение Golden Eye, а именно ручное изменение курсов валют с сайта и построение системы защиты.
- Для ручного изменения курсов необходимо добавить новую html страницу, на которой можно будет ввести новое значение для выбранного курса валют и сохранить внесенные изменения. Как правило в Web разработке это реализовывается с помощью html форм, причем запрос на сохранение требует HTTP запроса типа POST.
- Существует несколько типов HTTP запросов, все запросы, которые мы обрабатывали до этого — это были GET запросы. Как правило, они используются для получения и(или) отображения информации — что мы собственно и делали. Для добавления новой информации и(или) обновления существующей принято использовать HTTP методы PUT и POST. Основное отличие от GET запросов это то, что в теле запроса могут быть переданы собственно новые данные. В случае GET некоторые данные могут быть переданы в параметрах url (как мы указывали для фильтрации нужных курсов)

CyberBionic systematics
CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

- Flask поддерживает обработку различных HTTP методов. Для указания ожидаемого метода необходимо указать параметр methods (список) при вызове метода route при указании url. По умолчанию все запросы ожидаются как GET.
- Для обработки запроса по сохранению курсов валют нужно будет использовать метод POST. Для рендеринга шаблона с формой ввода данных о курсе — метод GET. Можно "повесить" эти методы один и тот же url: /edit/<from_curr>/<to_curr>, указав в параметре methods = ["GET", "POST"] два метода. А при обработке запроса, в зависимости от метода производить разные действия — либо просто отображать html страницу с формой (GET запрос), либо производить сохранение данных о курсе в БД (POST запрос).
- Помимо метода запроса (request.method) и уже известном нам request.args, в глобальном объекте request Flask запроса содержится много интересной информации — заголовки запроса (request.headers), ip адрес клиента (request.remote_addr), реферер — url, с которого пользователь был перенаправлен на текущую страницу — request.referrer, а также словарь ImmutableMultiDict с данными POST запроса — request.form. При доступе по ключу к словарю request.form, если ключ на найден в словаре, будет возвращаться ошибка Bad Request.
- Добавление нового контроллера для обработки запросов GET и POST при обновлении данных о курсе валют. Небольшие изменения в базовом контроллере для сохранения кода ошибки. Проверка в браузере.
- Необходимость ограничения доступа к странице с обновлением выбранного курса для всех пользователей Интернет, необходимость в простейшей системе защиты.
- Для реализации системы защиты, выберем механизм декораторов в Python.
- Декоратор — это паттерн проектирования, который позволяет динамически добавлять новую функциональность к объекту, но не путем наследования. В Python декораторы интересны тем, что реализованы на уровне синтаксиса языка и представляют собой очень полезный инструмент.
- Благодаря тому, что в Python все является объектом, в том числе и функция, и функции могут является аргументами других функций, можно декорировать функции дополнительным функционалом других функций. Подробнее рассмотрим на простейших примерах.
- Для организации простейшей системы защиты некоторых страниц, давайте добавим проверку по ip — то есть страница с отображением html формы с вводом нового курса валют и обновление курса в БД будет доступно только для ip владельцев сайта.
- Новый функционал, который мы хотим добавить к нашим некоторым view-функциям — это проверка, что ip адрес запроса находится в списке разрешенных ip адресов. Список с разрешенными ip адресами добавим в конфигурацию приложения и напишем новую функцию, которая будет осуществлять проверку. Если ip адрес не из списка разрешенных, возвращать ошибку 403 с помощью Flask функции abort. А затем просто декорируем нужные нам view-функции и проверим в браузере.

Закрепление материала

- В чем основное отличие методов HTTP — GET и POST?
- Как получить параметры url запроса во Flask?
- Как получить данные POST запроса во Flask?
- Как получить значение конкретного input поля формы при обработке POST запроса во Flask?
- Какие полезные атрибуты есть у объекта Flask запроса?
- Что такое декораторы и как они используются в Python?

Дополнительное задание

Задание
Добавить сохранение поля updated у курса при обновлении данных в БД.

CyberBionic systematics
CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Самостоятельная деятельность учащегося

Задание 1
Добавить в коде проверку, что курс с указанными from_currency и to_currency существует в БД. Если нет — генерировать ошибку 404 Not Found. Проверить в браузере.

Задание 2
Написать декоратор для, который будет выводить в консоль информацию о запросе — ip, заголовки, реферер, а также HTTP код ответа и декорировать им все view-функции.

Рекомендуемые ресурсы

https://www.w3schools.com/html/html_forms.asp
https://ru.wikipedia.org/wiki/POST_(HTTP)
http://flask.pocoo.org/docs/1.0/api/#flask.Request
http://flask.pocoo.org/docs/1.0/api/#flask.Request.form
https://habr.com/ru/post/141411/
http://flask.pocoo.org/docs/1.0/api/#flask.abort
https://docs.python.org/3/library/functools.html#functools.wraps
http://flask.pocoo.org/docs/0.12/patterns/viewdecorators/

CyberBionic systematics
CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com