Open-Source Technology Use Report

HTML Multipart Forms

General Information & Licensing

Code Repository	https://github.com/pallets/flask.git https://github.com/pallets/werkzeug.git https://github.com/pallets/werkzeug/blob/main/src/werkzeug/wrappers/request.py https://github.com/pallets/werkzeug/blob/main/src/werkzeug/sansio/request.py
License Type	BSD 3-Clause
License Description	• permissive license requiring redistributions in source and binary forms (with and without modification) to retain the copyright notice • the names of the copyright owner or contributors may not be used to endorse or promote products built from the library
License Restrictions	• Does not provide any warranty or liability
Who worked with this?	Ahsan

request.get (method)

Purpose

This library function allows for HTML multipart form parsing to get specified values that are helpful for HTML forms used for logging in or registering flask_server.py – lines 45 to 47: - this is where the request class is invoked and the “username” and “password” values from the HTML form are parsed and retrieved as strings

Magic ⋆★˚₊ °·°☽°⁀➷｡°★彡⋆ ༄

- The way that this library works when you look at the bigger picture is by using multipart HTML form parsing
- First, the *request* class (https://github.com/pallets/werkzeug/blob/main/src/werkzeug/wrappers/request.py) is invoked and the *form* method is called
    - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/wrappers/request.py#L413
- The request class also creates a Request object
    - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/sansio/request.py#L344
- In turn, the *form* method calls the *_load_form_data* method
    - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/wrappers/request.py#L251
- In turn, the *_load_form_data* method calls the *parse* method in *parser.py*
    - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/formparser.py#L230
- The *parse* method calls the *MultipartDecoder* in *multipart.py*, in which the actual HTTP request parsing and setting of headers is done
    - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/sansio/multipart.py#L67
- Next, the *_parse_multipart* method is called after the mime type, boundary, and other headers of the Request object are set
    - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/formparser.py#L271
        - this sets the fields variable to the key passed in in the initial call on line 45 or 47 of *flask_server.py* (all the way back to our own code)
            - https://github.com/pallets/werkzeug/blob/c7ae2fea4fb229ffd71187c2b665874c91b96277/src/werkzeug/formparser.py#L455
                - and lastly, a list of the values associated with the given multipart form field (such as “username” or “password”) is returned as part of this function, or the first value if given normal dictionary syntax, which the initial *get* method call does use in *flask.server.py* at lines 45 and 47