```
"""
Example script showing how to represent todo lists and todo entries in Python
data structures and how to implement endpoint for a REST API with Flask.

Requirements:
* flask
"""

import uuid

from flask import Flask, request, jsonify, abort


# initialize Flask server
app = Flask(__name__)

# create unique id for lists, entries
todo_list_1_id = '1318d3d1-d979-47e1-a225-dab1751dbe75'
todo_list_2_id = '3062dc25-6b80-4315-bb1d-a7c86b014c65'
todo_list_3_id = '44b02e00-03bc-451d-8d01-0c67ea866fee'
todo_1_id = uuid.uuid4()
todo_2_id = uuid.uuid4()
todo_3_id = uuid.uuid4()
todo_4_id = uuid.uuid4()

# define internal data structures with example data
todo_lists = [
    {'id': todo_list_1_id, 'name': 'Einkaufsliste'},
    {'id': todo_list_2_id, 'name': 'Arbeit'},
    {'id': todo_list_3_id, 'name': 'Privat'},
]
todos = [
    {'id': todo_1_id, 'name': 'Milch', 'description': '', 'list':
todo_list_1_id},
    {'id': todo_2_id, 'name': 'ArbeitsblÃ¤tter ausdrucken', 'description': '',
'list': todo_list_2_id},
    {'id': todo_3_id, 'name': 'Kinokarten kaufen', 'description': '', 'list':
todo_list_3_id},
    {'id': todo_3_id, 'name': 'Eier', 'description': '', 'list':
todo_list_1_id},
]

# add some headers to allow cross origin access to the API on this server,
necessary for using preview in Swagger Editor!
@app.after_request
def apply_cors_header(response):
    response.headers['Access-Control-Allow-Origin'] = '*'
    response.headers['Access-Control-Allow-Methods'] = 'GET,POST,DELETE'
    response.headers['Access-Control-Allow-Headers'] = 'Content-Type'
    return response

# define endpoint for getting and deleting existing todo lists
@app.route('/list/<list_id>', methods=['GET', 'DELETE'])
def handle_list(list_id):
    # find todo list depending on given list id
    list_item = None
    for l in todo_lists:
        if l['id'] == list_id:
            list_item = l
            break
    # if the given list id is invalid, return status code 404
```

```
    if not list_item:
        abort(404)
    if request.method == 'GET':
        # find all todo entries for the todo list with the given id
        print('Returning todo list...')
        return jsonify([i for i in todos if i['list'] == list_id])
    elif request.method == 'DELETE':
        # delete list with given id
        print('Deleting todo list...')
        todo_lists.remove(list_item)
        return '', 200


# define endpoint for adding a new list
@app.route('/list', methods=['POST'])
def add_new_list():
    # make JSON from POST data (even if content type is not set correctly)
    new_list = request.get_json(force=True)
    print('Got new list to be added: {}'.format(new_list))
    # create id for new list, save it and return the list with id
    new_list['id'] = uuid.uuid4()
    todo_lists.append(new_list)
    return jsonify(new_list), 200


# define endpoint for getting all lists
@app.route('/lists', methods=['GET'])
def get_all_lists():
    return jsonify(todo_lists)


if __name__ == '__main__':
    # start Flask server
    app.debug = True
    app.run(host='0.0.0.0', port=5000)
```