Python flask.request.get_json() Examples

The following are code examples for showing how to use *flask.request.get_json()*. They are from open source Python projects. You can vote up the examples you like or vote down the ones you don't like.

Example 1

Project: *BASS* Author: *Cisco-Talos* File: *server.py* GNU General Public License v2.0 7 vo

```
def whitelist_add():
    log.info("whitelist_add called")
    try:
        file_ = request.files["file"]
        handle, filename = tempfile.mkstemp()
        os.close(handle)
        file_.save(filename)
        data = request.get_json()
        if data and "functions" in data:
            functions = data["functions"]
        else:
            functions = None
        bass.whitelist_add(filename, functions)
        os.unlink(filename)
    except KeyError:
        log.exception("")
        return make_response(jsonify(message = "Sample file 'file' missing in POST

    return jsonify(message = "OK")
```

Example 2

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin003.py* Apache License 2.0 7 vo

```
def add_transaction():
    # getting the data from a separate json file.
    json = request.get_json()

    # the keys that should be included in the json file.
    transaction_keys = ['sender' , 'receiver', 'amount']

    # return a error message if a key is not included in the file.
    if not all (key in json for key in transaction_keys):
        return 'Something Missing' , 400

    # getting the data from the json file.
    index = blockchain.add_transaction(json[transaction_keys[0]],
                                       json[transaction_keys[1]],
                                       json[transaction_keys[2]])

    # output back the message of succes
    response = {'message' : f'This transactoin will be added to block {index}'}
    return response , 201

# Part 3 - Decentralizing Blockchain

# Connecting new nodes
```

Example 3

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin.py* Apache License 2.0 6 vo

```
def add_transaction():
    # getting the data from a separate json file.
    json = request.get_json()

    # the keys that should be included in the json file.
    transaction_keys = ['sender' , 'receiver', 'amount']

    # return a error message if a key is not included in the file.
    if not all (key in json for key in transaction_keys):
        return 'Something Missing' , 400

    # getting the data from the json file.
    index = blockchain.add_transaction(json[transaction_keys[0]],
                                       json[transaction_keys[1]],
                                       json[transaction_keys[2]])

    # output back the message of succes
    response = {'message' : f'This transactoin will be added to block {index}'}
    return response , 201

# Part 3 - Decentralizing Blockchain

# Connecting new nodes
```

Example 4

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin002.py* Apache License 2.0 6 vo

```
def add_transaction():
    # getting the data from a separate json file.
    json = request.get_json()

    # the keys that should be included in the json file.
    transaction_keys = ['sender' , 'receiver', 'amount']

    # return a error message if a key is not included in the file.
    if not all (key in json for key in transaction_keys):
        return 'Something Missing' , 400

    # getting the data from the json file.
    index = blockchain.add_transaction(json[transaction_keys[0]],
                                       json[transaction_keys[1]],
                                       json[transaction_keys[2]])

    # output back the message of succes
    response = {'message' : f'This transactoin will be added to block {index}'}
    return response , 201

# Part 3 - Decentralizing Blockchain

# Connecting new nodes
```

Example 5

Project: *radius-1xtest* Author: *shanghai-edu* File: *views.py* Apache License 2.0 6 vo

```
def radius1x_api(ssid):
        ssid_config = app.config['SSID_CONFIG']
        if ssid not in ssid_config:
```

```
                return "404 page not found",404
        data = request.get_json()
        if not data or not 'username' in data or not 'password' in data:
                result = {"success":False,"msg":"username or password not found"}
                return jsonify(result=result),400
        username = data["username"].encode("utf-8")
        password = data["password"].encode("utf-8")
        tsStart = time()
        try:
                res = radius_challenge(username, password, ssid_config[ssid]['RADI
                t = time() - tsStart
                result = {"username":username,"method":"mscharpv2","time":t,"succe
                return jsonify(result=result),200
        except:
                t = time() - tsStart
                result = {"username":username,"method":"mscharpv2","time":t,"succe
                return jsonify(result=result),200
```

Example 6

Project: *bounty_tools* Author: *gradiuscypher* File: *reconng.py* MIT License 6 vo

```
def run():

    # Setup the jsonrpclib for the recon-ng RPC server, stop the API if it cannot
    try:
        client = jsonrpclib.Server('http://localhost:4141')
        sid = client.init()

        # Get the configuration from JSON POST
        content = request.get_json()
        target_module = content['module']
        target_domain = content['domain']
        print(target_domain, target_module)

        # Set the target domain
        client.add('domains', target_domain, sid)
        print(client.show('domains', sid))
        client.use(target_module, sid)

        # Execute the requested module and return the results
        results = client.run(sid)

        return jsonify(results)

    except:
        return traceback.format_exc(), 500
```

Example 7

Project: *DancesafeResults* Author: *siorai* File: *DancesafeResults.py* GNU Affero General Public License v3.0 6 vo

```
def api_new_event():
    if request.method == "POST":
        newConnection = DBConn()
        currentSession = newConnection.cursor()
        print(request.is_json)
        content = request.get_json()
        print(content)
        eventName = content["name"]
```

```
        eventYear = content["year"]
        eventCity = content["city"]
        eventState = content["state"]
        eventRegion = content["region"]
        eventAuthor = content["author"]
        currentSession.execute(
            "INSERT INTO EVENT (name, year, city, state, region, author) VALUES ('
                eventName, eventYear, eventCity, eventState, eventRegion, eventAut
            )
        )
        newConnection.commit()
        currentSession.close()
        newConnection.close()
        return "JSON posted"
```

Example 8

Project: *geo-knowledge-hub* Author: *geosec* File: *views.py* MIT License 6 vo

```
def login():
    form = request.get_json() or {}

    #email = form.get('email')

    with db.session.begin_nested():
        user = User.query.first()

        if not user:
            user = User()
            user.email = 'admin@invenio.org'
            user.active = True
            user.password = '123456'

            db.session.add(user)

    db.session.commit()

    login_user(user, remember=True)

    return jsonify({'status': 'ok', 'sessionid': session['_id']})
```

Example 9

Project: *CTask* Author: *yangmv* File: *views.py* GNU General Public License v3.0 6 vo

```
def edit_job():
    '''修改作业'''
    response = {'status': '-1'}
    try:
        data = request.get_json(force=True)
        job_id = data.get('id')
        old_job = scheduler.get_job(job_id)
        if old_job:
            jobfromparm(scheduler,**data)
            response['status'] = 0
            response['message'] = "job[%s] edit success!"%job_id
        else:
            response['message'] = "job[%s] Not Found!"%job_id
    except Exception as e:
        response['message'] = str(e)
    return json.dumps(response)
```

Example 10

Project: *PyCoin* Author: *NovemberOscar* File: *server.py* MIT License 6 vo

```
def register_nodes():

    values = request.get_json()

    nodes = values.get('nodes')
    print(nodes)

    if nodes is None or type(nodes) == str:
        return "Error : unvalid list of nodes", 400

    for node in nodes:
        blockchain.register_node(node)

    response = {
        'message': 'New Nodes hav been successfully added',
        'total_nodes': list(blockchain.nodes)
    }

    return jsonify(response), 201
```

Example 11

Project: *Garnet* Author: *OneTesseractInMultiverse* File: *account_controller.py* MIT License 6 vo

```
def update_account_password(user_id):
    try:
        pass_data = request.get_json()
        user_service = UserService(user_id)
        usr = user_service.get_user()
        if user_id == current_identity.id:
            if usr.update_password(pass_data['password']):
                app.logger.info('Updated password for user_id: %s', user_id)
                return SuccessResponse('Success', 'Password updated successfully',
        else:
            app.logger.error('Permission violation. User not authorized to update
            return ErrorResponse('Permission violation', 'This action generated a
    except:
        app.logger.error('Invalid json received for user: %s', user_id)
        return ErrorResponse('Could not update password', 'Invalid password provic


# ------------------------------------------------------------------------
# PUT: /account/<uid>/email
# ------------------------------------------------------------------------
```

Example 12

Project: *Garnet* Author: *OneTesseractInMultiverse* File: *account_controller.py* MIT License 6 vo

```
def update_account_email(user_id):
    try:
        email_data = request.get_json()
        user_service = UserService(user_id)
        user = user_service.get_user()
        if user.update_email(email_data['email']):
            app.logger.info('Updated email for user_id: %s', user_id)
            return SuccessResponse('Success', 'Email updated successfully', 'EMAII
```

```
    except:
        app.logger.error('Invalid json received for user: %s', user_id)
        return ErrorResponse('Could not update email', 'Invalid email provided').a


# ------------------------------------------------------------------------
# POST: /account
# ------------------------------------------------------------------------
# Registers a new user in the system using garnet_api Identity Sub-System
```

Example 13

Project: *Garnet* Author: *OneTesseractInMultiverse* File: *account_controller.py* MIT License 6 vo

```
def post_account():
    user_data = request.get_json()
    if user_data:
        user = User(
            user_id=str(uuid.uuid4()),
            name=user_data['name'],
            last_name=user_data['last_name'],
            email=user_data['email'],
            username=user_data['username'],
            password=None
            )
        user.update_password(user_data['password'])
        user.save(validate=True)
        app.logger.info('User %s was created', user.user_id)
        return SuccessResponse(user.user_id, 'User created successfully', 'n/a').a
    return ErrorResponse('Error processing request', 'The provided data is not val
```

Example 14

Project: *dino* Author: *thenetcircle* File: *routes.py* Apache License 2.0 6 vo

```
def create_channel():
    """ Create new channel """
    form = request.get_json()
    channel_name = form['name']
    channel_uuid = str(uuid())
    user_uuid = form['owner']

    message = {}
    if is_blank(channel_name):
        message['name'] = "Channel name can't be none."
    if is_blank(user_uuid):
        message['owner'] = "Owner can't be none."

    if len(message):
        return api_response(400, message=message)
    result = channel_manager.create_channel(channel_name, channel_uuid, user_uuid)

    if result is not None:
        return api_response(400, message=result)
    return api_response(200, {'sort': 1, 'name': channel_name, 'uuid': channel_uui
```

Example 15

Project: *dino* Author: *thenetcircle* File: *routes.py* Apache License 2.0 6 vo

```
def update_channel_acl(channel_uuid: str, action: str, acl_type: str):
    form = request.get_json()
    value = form['value']

    try:
        acl_manager.update_channel_acl(channel_uuid, action, acl_type, value)
    except InvalidAclValueException:
        return api_response(400, message='Invalid ACL value %s' % value)
    except InvalidAclTypeException:
        return api_response(400, message='Invalid ACL type %s' % acl_type)
    except ValidationException as e:
        return api_response(400, message='Invalid ACL: %s' % e.msg)
    except Exception as e:
        logger.exception(traceback.format_exc())
        return api_response(400, message='could not update acl for channel %s: %s'

    return api_response(200)
```

Example 16

Project: *dino* Author: *thenetcircle* File: *routes.py* Apache License 2.0 6 vo

```
def update_room_acl(channel_uuid: str, room_uuid: str, action: str, acl_type: str)
    form = request.get_json()
    value = form['value']

    try:
        acl_manager.update_room_acl(channel_uuid, room_uuid, action, acl_type, val
    except InvalidAclValueException:
        return api_response(400, message='Invalid ACL value %s' % value)
    except InvalidAclTypeException:
        return api_response(400, message='Invalid ACL type %s' % acl_type)
    except ValidationException as e:
        return api_response(400, message='Invalid ACL: %s' % e.msg)
    except Exception as e:
        logger.exception(traceback.format_exc())
        return api_response(400, message='could not update acl for room %s: %s' %

    return api_response(200)
```

Example 17

Project: *dino* Author: *thenetcircle* File: *routes.py* Apache License 2.0 6 vo

```
def send_broadcast():
    form = request.get_json()
    verb = form['verb']
    content = form['content']

    message = {}
    if is_blank(verb):
        message['verb'] = 'Verb may not be empty.'
    if is_blank(content):
        message['content'] = 'Content may not be empty.'

    if len(message):
        return api_response(400, message=message)

    try:
        content = utils.b64e(content)
        broadcast_manager.send(content, verb)
```

```
    except Exception as e:
        logger.error('Could not send broadcast: %s' % str(e))
        logger.exception(traceback.format_exc())
        return api_response(400, message='Could not send broadcast')
    return api_response(200)
```

Example 18

Project: *beavy* Author: *beavyHQ* File: *views.py* Mozilla Public License 2.0 6 vo

```
def submit_story():
    if request.method == "POST":
        params = request.get_json()
        title, url = params['title'].strip(), params['url'].strip()
        text = params.get('text', "").strip()
        if not title:
            return abort(400, "You have to provide a 'title'")

        if url:
            link = Link(title=title, url=url, owner_id=current_user.id)
            db.session.add(link)
            db.session.commit()
            return link_schema.dump(link)
        elif text:
            topic = Topic(title=title, text=text, owner_id=current_user.id)
            db.session.add(topic)
            db.session.commit()
            return topic_schema.dump(topic)

        return abort(400, "You have to provide either 'url' or 'text', too")

    # Just render it
    return {}
```

Example 19

Project: *plan-write-revise* Author: *seraphinatarrant* File: *web_server.py* MIT License 6 vo

```
def write_auto_txt():
    request_json = request.get_json(force=True)
    with open("auto_mode_logging.txt", "a") as out:
        out.write("New auto generation data:" + "\n")
        out.write("Story Topic:" + "\n")
        out.write(request_json.get('topic') + "\n")
        out.write("Storyline:" + "\n")
        out.write(request_json.get('storyline') + "\n")
        out.write("System1_story:" + "\n")
        out.write(request_json.get('system1_story') + "\n")
        out.write("System2_story:" + "\n")
        out.write(request_json.get('system2_story') + "\n")
        out.write("System3_story:" + "\n")
        out.write(request_json.get('system3_story') + "\n")

    return "success"


# write interactive mode data to txt
```

Example 20

5 vo

Project: *telegram-innovation-chatbot* Author: *zaoldyeck* File: *main.py* MIT License

```
def webhook_handler():
    """Set route /hook with POST method will trigger this method."""
    if request.method == "POST":
        update = telegram.Update.de_json(request.get_json(force=True), bot)
        dispatcher.process_update(update)
    return 'ok'
```

Example 21

Project: *oscap-daemon-api* Author: *mvazquezc* File: *api.py* GNU Lesser General Public License v2.1 5 vo

```
def newTask():
    content = request.get_json(silent=False)
    requiredFields = {'taskTitle', 'taskTarget', 'taskSSG', 'taskTailoring', 'task
    if content is None:
        return '{ "Error" : "json data required" }', 400
    elif not requiredFields <= set(content):
        return '{ "Error": "There are missing fields in the request" }', 400
    elif content['taskSSG'] == "" or content['taskProfileId'] == "":
        return '{ "Error": "Both taskSSG and taskProfileId fields cannot be empty'
    else:
        response = oscapd.new_task(content['taskTitle'], content['taskTarget'],
            content['taskSSG'], content['taskTailoring'], content['taskProfileId']
            content['taskOnlineRemediation'], content['taskScheduleNotBefore'],
            content['taskScheduleRepeatAfter'])
        return response, 201
```

Example 22

Project: *oscap-daemon-api* Author: *mvazquezc* File: *api.py* GNU Lesser General Public License v2.1 5 vo

```
def updateTask(taskId):
    content = request.get_json(silent=False)
    requiredFields = {'taskTitle', 'taskTarget', 'taskSSG', 'taskTailoring', 'task
    if content is None:
        return '{ "Error" : "json data required" }', 400
    elif not requiredFields <= set(content):
        return '{ "Error": "There are missing fields in the request" }', 400
    else:
        response = oscapd.update_task(taskId, content['taskTitle'], content['taskT
            content['taskSSG'], content['taskTailoring'], content['taskProfileId']
            content['taskOnlineRemediation'], content['taskScheduleNotBefore'],
            content['taskScheduleRepeatAfter'])
        return response
```

Example 23

Project: *oscap-daemon-api* Author: *mvazquezc* File: *api.py* GNU Lesser General Public License v2.1 5 vo

```
def getSSG():
    content = request.get_json(silent=False)
    requiredFields = {'ssgFile', 'tailoringFile' }
    if content is None:
        return '{ "Error" : "json data required" }', 400
    elif not requiredFields <= set(content):
        return '{ "Error": "There are missing fields in the request" }', 400
    elif content['ssgFile'] == "":
        return '{ "Error": "ssgFile field cannot be empty" }', 400
```

```
    else:
        response = oscapd.get_ssg(content['ssgFile'],content['tailoringFile'])
        return response
```

Example 24

Project: *reroils-data-legacy* Author: *rero* File: *utils.py* GNU General Public License v2.0 5 vo

```
def item_from_web_request(data):
    """Get item from web request data."""
    data = request.get_json()
    pid = data.pop('pid')
    return Item.get_record_by_pid(pid)
```

Example 25

Project: *pnp* Author: *HazardDede* File: *http.py* MIT License 5 vo

```
def _create_app(self):
        that = self
        flask = load_optional_module('flask', self.EXTRA)
        app = flask.Flask(__name__)

        if self.server_impl == 'flask':
            # We need to register a shutdown endpoint, to end the serving if using
            # development server
            @app.route('/_shutdown', methods=['DELETE'])
            def shutdown():  # pylint: disable=unused-variable
                from flask import request
                func = request.environ.get('werkzeug.server.shutdown')
                if func is None:
                    raise RuntimeError('Not running with the Werkzeug Server')  #
                func()
                return json.dumps({'success': True}), 200, {'ContentType': 'applic

        @app.route('/', defaults={'path': '/'}, methods=self.allowed_methods)
        @app.route('/<path:path>', methods=self.allowed_methods)
        def catch_all(path):  # pylint: disable=unused-variable
            from flask import request
            data = request.get_json(force=True, silent=True)
            if data is None:  # No valid json in request body > fallback to data
                data = request.data if request.data != b'' else None

            payload = dict(
                endpoint=path,
                levels=["/"] if path == "/" else path.split('/'),
                method=request.method,
                query=self._flatten_query_args(dict(request.args)),
                data=data,
                is_json=isinstance(data, dict),
                url=request.url,
                full_path=request.full_path,
                path=request.path
            )
            that.notify(payload)

            return json.dumps({'success': True}), 200, {'ContentType': 'applicatio

        return app
```

Example 26

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin003.py* Apache License 2.0 5 vo

```
def connect_node():
    json = request.get_json()
    nodes = json.get('nodes')
    if nodes is  None:
        return "No nodes",400
    for node in nodes:
        blockchain.add_node(node)
    response = {'message' : 'All the node are connected' ,
                'total_nodes' : list(blockchain.nodes)}
    return jsonify(response),200

# Replacing by the new longest chain.
```

Example 27

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin.py* Apache License 2.0 5 vo

```
def connect_node():
    json = request.get_json()
    nodes = json.get('nodes')
    if nodes is  None:
        return "No nodes",400
    for node in nodes:
        blockchain.add_node(node)
    response = {'message' : 'All the node are connected' ,
                'total_nodes' : list(blockchain.nodes)}
    return jsonify(response),200

# Replacing by the new longest chain.
```

Example 28

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin001.py* Apache License 2.0 5 vo

```
def add_transaction():
    # getting the data from a separate json file.
    json = request.get_json()

    # the keys that should be included in the json file.
    transaction_keys = ['sender' , 'receiver', 'amount']

    # return a error message if a key is not included in the file.
    if not all (key in json for key in transaction_keys):
        return 'Something Missing' , 400

    # getting the data from the json file.
    index = blockchain.add_transaction(json[transaction_keys[0]],
                                       json[transaction_keys[1]],
                                       json[transaction_keys[2]])

    # output back the message of succes
    response = {'message' : f'This transactoin will be added to block {index}'}
    return response , 201

# Part 3 - Decentralizing Blockchain

# Connecting new nodes
```

Example 29

Project: *Blockchain* Author: *Younes-Charfaoui* File: *chacoin001.py* Apache License 2.0 5 vo

```
def connect_node():
    json = request.get_json()
    nodes = json.get('nodes')
    if nodes is  None:
        return "No nodes",400
    for node in nodes:
        blockchain.add_node(node)
    response = {'message' : 'All the node are connected' ,
                'total_nodes' : list(blockchain.nodes)}
    return jsonify(response),200

# Replacing by the new longest chain.
```

Example 30

Project: *Bluemix-ServiceBroker* Author: *IBM-Cloud* File: *bmx-sample-broker.py* Apache License 2.0 5 vo

```
def provision(instance_id):
    # Provision an instance of this service for the org/space
    # as provided in the JSON data
    #
    # PUT /v2/service_instances/<instance_id>:
    #    <instance_id> provided by Bluemix Cloud Controller,
    #   used for future requests like bind, unbind and deprovision
    #
    # BODY:
    #     {
    #       "service_id":        "<service-guid>",
    #       "plan_id":           "<plan-guid>",
    #       "organization_guid": "<org-guid>",
    #       "space_guid":        "<space-guid>"
    #     }
    #
    # return:
    #     JSON document with service details

    if request.headers['Content-Type'] != 'application/json':
        abort(415, 'Unsupported Content-Type: expecting application/json')
    # get the JSON document in the BODY
    provision_details = request.get_json(force=True)

    # provision the service by calling out to the service itself
    # not done here to keep the code simple for the tutorial

    # return basic service information
    new_service={"dashboard_url": service_dashboard+instance_id}
    return jsonify(new_service)


#
# Deprovision
#
```

Example 31

Project: *Bluemix-ServiceBroker* Author: *IBM-Cloud* File: *bmx-sample-broker.py* Apache License 2.0 5 vo

```
def bind(instance_id, binding_id):
    # Bind an existing instance with the given org and space
```

```
    #
    # PUT /v2/service_instances/<instance_id>/service_bindings/<binding_id>:
    #     <instance_id> is the Cloud Controller provided
    #       value used to provision the instance
    #     <binding_id> is provided by the Cloud Controller
    #       and will be used for future unbind requests
    #
    # BODY:
    #     {
    #       "plan_id":           "<plan-guid>",
    #       "service_id":        "<service-guid>",
    #       "app_guid":          "<app-guid>"
    #     }
    #
    # return:
    #     JSON document with credentails and access details
    #     for the service based on this binding
    #     http://docs.cloudfoundry.org/services/binding-credentials.html

    if request.headers['Content-Type'] != 'application/json':
        abort(415, 'Unsupported Content-Type: expecting application/json')

    # get the JSON document in the BODY
    binding_details = request.get_json()

    # bind would call the service here
    # not done to keep our code simple for the tutorial


    # return result to the Bluemix Cloud Controller
    result={"credentials": {"uri": "testme"}}
    return make_response(jsonify(result),201)

#
# Unbind
#
```

Example 32

Project: *PickTrue* Author: *winkidney* File: *taskserver.py* MIT License 5 vo

```
def task_submit():
    """
    :return:
    """
    resp = request.get_json(force=True)
    server.requester.submit_response(
        resp
    )
    return jsonify({})
```

Example 33

Project: *bendercoin* Author: *matejcik* File: *bank.py* MIT License 5 vo

```
def send_tx():
    data = request.get_json(force=True)
    tx = Transaction.from_dict(data)

    try:
        transact(tx)
```

```
        return jsonify(status="ok")
    except Exception as e:
        return jsonify(status="err", error=str(e))
```

Example 34

Project: *cis* Author: *mozilla-iam* File: *api.py* Mozilla Public License 2.0 5 vo

```
def change():
    connection = connect.AWS()
    connection.session()
    identity_vault_client = connection.identity_vault_client()

    user_profile = request.get_json(silent=True)
    if isinstance(user_profile, str):
        user_profile = json.loads(user_profile)

    user_id = request.args.get("user_id", user_profile["user_id"]["value"])
    logger.info("A json payload was received for user: {}".format(user_id), extra=
    vault = profile.Vault(sequence_number=None, profile_json=user_profile, **reque

    if request.method in ["POST", "PUT", "GET"]:
        vault.identity_vault_client = identity_vault_client
        result = vault.put_profile(user_profile)
        logger.info(
            "The result of publishing for user: {} is: {}".format(user_id, result)
            extra={"user_id": user_id, "result": result},
        )
    if config("allow_delete", namespace="cis", default="false") == "true":
        if request.method in ["DELETE"]:
            vault.identity_vault_client = identity_vault_client
            result = vault.delete_profile(user_profile)
            logger.info(
                "A delete operation was performed for user: {}".format(user_id),
                extra={"user_id": user_id, "result": result},
            )
    return jsonify(result)
```

Example 35

Project: *cis* Author: *mozilla-iam* File: *api.py* Mozilla Public License 2.0 5 vo

```
def changes():
    connection = connect.AWS()
    connection.session()
    identity_vault_client = connection.identity_vault_client()
    profiles = request.get_json(silent=True)
    logger.info("A list numbering: {} profiles has been received.".format(len(prof
    vault = profile.Vault(sequence_number=None)
    vault.identity_vault_client = identity_vault_client
    results = vault.put_profiles(profiles)
    logger.info("The result of the attempt to publish the profiles was: {}".format
    return jsonify(results)
```

Example 36

Project: *attack-graphs* Author: *cyberImperial* File: *components.py* MIT License 5 vo

```
def receive_post(self):
        if request.method == "POST":
```

```
            req = request.get_json()
            output = self.process(req)
            return str(output)
```

Example 37

Project: *DancesafeResults* Author: *siorai* File: *DancesafeResults.py* GNU Affero General Public License v3.0 5 vo

```
def api_add_sample():
    if request.method == "POST":
        newConnection = DBConn()
        currentSession = newConnection.cursor()
        print(request.is_json)
        content = request.get_json()
        print(content)
        eventid = content["eventid"]
        shiftLead = content["shiftLead"]
        tester = content["tester"]
        recorder = content["recorder"]
        typeid = content["typeid"]
        initialSuspect = content["initialSuspect"]
        description = content["description"]
        groundscore = content["groundscore"]
        conclusiveResult = content["conclusiveResult"]
        finalConclusion = content["finalConclusion"]
        acquiredOnSite = content["acquiredOnSite"]
        planToIngest = content["planToIngest"]
        currentSession.execute(
            "INSERT INTO SAMPLE (eventid, shiftlead, tester, recorder, typeid, ini
                eventid,
                shiftLead,
                tester,
                recorder,
                typeid,
                initialSuspect,
                description,
                groundscore,
                conclusiveResult,
                finalConclusion,
                acquiredOnSite,
                planToIngest,
            )
        )
        newConnection.commit()
        currentSession.close()
        newConnection.close()
        return "JSON posted"
```

Example 38

Project: *warbadge* Author: *robotlandman* File: *app.py* MIT License 5 vo

```
def update_handle(badge_mac):
    """ This route creates a new entry for a handle to badge mac
    mapping.
    """
    log.info("update handle for %s", badge_mac)
    request_json = request.get_json()
    insert_template = (u"INSERT INTO handles (badge_mac, handle) "
                       "VALUES('{0}', '{1}')"
```

```
                    .format(badge_mac, request_json['handle']))
    update_template = (u"UPDATE handles SET handle = '{1}' WHERE badge_mac = '{0}'
                    .format(badge_mac, request_json['handle']))
    conn = mysql.connect()
    cursor = conn.cursor()
    try:
        cursor.execute(insert_template)
        conn.commit()
        log.debug("Finished a transaction: %s", insert_template)
        return_code = 201
    except IntegrityError as exception:
        if exception[0] == 1062:
            log.info("handle %s: already exists switching to update", badge_mac)
            try:
                cursor.execute(update_template)
                conn.commit()
                log.debug("Finished a transaction")
                return_code = 200
            except Exception as exception:  # pylint: disable=W0703
                log.warn("issue updating handle for %s: %s", badge_mac, exception)
                return_code = 409
        else:
            log.error("badge_mac %s: MySQL ERROR: %s", badge_mac, exception)
            log.error("MySQL ERROR: %s", exception)
            return_code = 500
    else:
        log.info("handle %s: added", badge_mac)
    finally:
        conn.close()
    payload = json.dumps({'warbadging': True})
    content_type = {'ContentType': 'application/json'}
    return payload, return_code, content_type
```

Example 39

Project: *ibart* Author: *jbech-linaro* File: *ibart.py* MIT License 5 vo

```
def dump_json_blob_to_file(request, filename="last_blob.json"):
    """ Debug function to dump the last json blob to file """
    with open(filename, 'w') as f:
        payload = request.get_json()
        json.dump(payload, f, indent=4)
```

Example 40

Project: *SayluaLegacy* Author: *saylua* File: *api.py* GNU Affero General Public License v3.0 5 vo

```
def api_send_score(game_id):
    try:
        gameName = Game(game_id)
    except IndexError:
        return json.dumps(dict(error='Invalid game!')), 400
    finally:
        if gameName == "blocks":
            # TODO sanity check the game log and other variables sent to catch
            # low hanging fruit attempts at cheating.
            data = request.get_json()
            score = int_or_none(data.get('score')) or 0
            GameLog.record_score(g.user.id, game_id, score)
            g.user.cloud_coins += score
            db.session.commit()
```

```
            return json.dumps(dict(cloud_coins=g.user.cloud_coins, star_shards=g.u
    return json.dumps(dict(error='Bad request.')), 400
```

Example 41

Project: *CTask* Author: *yangmv* File: *views.py* GNU General Public License v3.0 5 vo

```
def pause_job():
    '''暂停作业'''
    response = {'status': '-1'}
    try:
        data = request.get_json(force=True)
        job_id = data.get('id')
        scheduler.pause_job(job_id)
        response['msg'] = "job[%s] pause success!"%job_id
        response['status'] = 0
    except Exception as e:
        response['msg'] = str(e)
    return json.dumps(response)
```

Example 42

Project: *CTask* Author: *yangmv* File: *views.py* GNU General Public License v3.0 5 vo

```
def resume_job():
    '''恢复作业'''
    response = {'status': '-1'}
    try:
        data = request.get_json(force=True)
        job_id = data.get('id')
        scheduler.resume_job(job_id)
        response['msg'] = "job[%s] resume success!"%job_id
        response['status'] = 0
    except Exception as e:
        response['msg'] = str(e)
    return json.dumps(response)
```

Example 43

Project: *CTask* Author: *yangmv* File: *views.py* GNU General Public License v3.0 5 vo

```
def reomve_jobs():
    '''删除作业'''
    response = {'status': '-1'}
    try:
        data = request.get_json(force=True)
        job_id = data.get('id')
        if job_id != 'all':
            scheduler.remove_job(job_id)
            response['msg'] = "job[%s] remove success!"%job_id
        else:
            scheduler.remove_all_jobs()
            response['msg'] = "job all remove success!"
        response['status'] = 0
    except Exception as e:
        response['msg'] = str(e)
    return json.dumps(response)
```

Example 44

Project: *CTask* Author: *yangmv* File: *views.py* GNU General Public License v3.0 5 vo

```
def add_job():
    '''新增作业'''
    response = {'status': '-1'}
    try:
        data = request.get_json(force=True)
        job_id = jobfromparm(scheduler,**data)
        response['status'] = 0
        response['msg'] = "job[%s] add success!"%job_id
    except Exception as e:
        response['msg'] = str(e)
    return json.dumps(response)
```

Example 45

Project: *PyCoin* Author: *NovemberOscar* File: *server.py* MIT License 5 vo

```
def apply_account():

    values = request.get_json()

    account_id = values.get('id')
    print(account_id)

    if Account.apply_acount(account_id):
        response = jsonify({'message': 'ID was applied'}), 201
    else:
        response = jsonify({'message': 'requested ID is existing'}), 400

    return response
```

Example 46

Project: *PyCoin* Author: *NovemberOscar* File: *server.py* MIT License 5 vo

```
def new_transaction():
    values = request.get_json()
    print(values)

    check, index = blockchain.new_transactions(values['sender'], values['recipient

    if check:
        response = jsonify({'message': 'Transaction will be added to Block {0}'.fo
    else:
        response = jsonify({'message': 'Requested transaction is rejected'}), 403

    return response
```

Example 47

Project: *openc2-aws* Author: *att* File: *aws_netacl.py* BSD 2-Clause "Simplified" License 5 vo

```
def openc2_aws_sg():
    if request.headers['Content-Type'] == 'application/json':
        cmd = parse(request.get_json())
        try:
            naclap = AWSNACL(**cmd)
        except Exception as e:
            resp = Response(status=400,
```

```
            status_text="Invalid command format/arguments (%s)"%str(e))
            return resp.serialize()

        session = boto3.Session(profile_name=naclap.actuator.aws_account_id)
        ec2 = session.client('ec2',region_name=naclap.actuator.aws_region)

        try:
            if naclap.action == 'delete':
                data = ec2.delete_network_acl_entry(NetworkAclId=naclap.actuator.a
                    Egress=naclap.clean(naclap.args.slpf.direction, {'ingress':Fal
                    RuleNumber=naclap.target.slpf.rule_number)
            else:
                data = ec2.create_network_acl_entry(NetworkAclId=naclap.actuator.a
                    Egress=naclap.clean(naclap.args.slpf.direction, {'ingress':Fal
                    PortRange={ 'From': naclap.target.dst_port, 'To': naclap.targe
                    Protocol=naclap.clean(naclap.target.protocol,{'tcp':6,'udp':17
                    RuleAction=naclap.action, RuleNumber=naclap.args.slpf.insert_r
        except Exception as e:
            #todo: parse boto3 for http code and resp
            resp = Response(status=400,
                    status_text=str(e))
            return resp.serialize()
        else:
            resp = Response(status=200,
                results = {"x-aws-nacl":data})
        return resp.serialize()
    else:
        resp = Response(status=425,
                        status_text="Unsupported Media Type")
        return resp.serialize()
```

Example 48

Project: *PyCasa* Author: *py-ranoid* File: *server.py* MIT License 5 vo

```
def prime():
    """Endpoint to process incoming requests.
    (POST requests with JSON mapping 'obj' to command)"""
    cont = request.get_json()
    if cont is not None:
        trigger(command_string=cont['obj'])
    return jsonify({"SUCCESS":True})
```

Example 49

Project: *TheHiveHooks* Author: *TheHive-Project* File: *controllers.py* GNU Affero General Public License v3.0 5 vo

```
def webhook():
    event = request.get_json()
    event_name = capitalize('{}_{}'.format(event['objectType'], event['operation']
    app.logger.info('Emit {}: Root={}, Details={}'.format(event_name, event['rootI
    ee.emit(event_name, event)

    return json.dumps(event, indent=4, sort_keys=True)
```

Example 50

Project: *flask-api-template* Author: *bonzanini* File: *dummy.py* MIT License 5 vo

```
def post(self):
        """Return a HelloResult object"""
        reqs = request.get_json()
        if not reqs:
            raise JsonRequiredError()
        try:
            reqs['name']
            return HelloResult(name=reqs['name'])
        except KeyError:
            raise JsonInvalidError()
```