LU06.L01 - Autorisierung

resource/auth_resource.py

```
import json
from datetime import datetime, timedelta

import jwt
from Crypto.Hash import SHA256
from flask import make_response, current_app
from flask_restful import Resource, reqparse

from model.user import User


class AuthResource(Resource):
    def __init__(self):
        self.parser = reqparse.RequestParser()
        self.parser.add_argument('username', location='form',
default=None, help='username')
        self.parser.add_argument('password', location='form',
default=None, help='password')

    def post(self):
        args = self.parser.parse_args()
        user = self._get_user_by_username(args.username)
        if user is not None:
            hash = self._password_hash(args.password)
            if user.password == hash:
                token = self._make_token(user)
                return make_response(token, 200)

        return make_response('', 401)

    def _get_user_by_username(self, username):
        """
        reads a user by its username
        :param username:
        :return:
        """
        with open('./files/users.json', encoding='UTF-8') as file:
            users_dict = json.load(file)
            for data in users_dict:
                if data['username'] == username:
                    user = User(
                        username=data['username'],
                        password=data['password'],
                        role=data['role']
                    )
```

```
                    return user

            return None

    def _password_hash(self, password):
        """
        creates the password hash
        :param password:
        :return:
        """
        hash_object = SHA256.new(data=password.encode())
        return hash_object.hexdigest()

    def _make_token(self, user):
        """
        creates the signed and encrypted token for a user
        :param user:
        :return:
        """
        access = jwt.encode({
            'username': user.username,
            'userrole': user.role,
            'exp': datetime.utcnow() +
timedelta(minutes=current_app.config['TOKEN_DURATION'])
        },
            current_app.config['ACCESS_TOKEN_KEY']
        )
        return access
```

M321-LU06