Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Звіт

З лабораторної роботи № 2

«Серверна частина системи ConditionsController»

З дисципліни «Архітектура, аналіз та проектування програмного забезпечення»

Підготував:

Ст. гр. ПІ-15-1

Євченко Богдан

2018

Методи перевірки авторизації користувача та девайсів.

Метод перевіряє чи є в системі такий користувач з заданими поштою та паролем, авторизує користувача, якщо такий був знайдений.  
@basic\_auth.verify\_password  
**def** verify\_password(email, password):  
 user = User.query.filter\_by(email=email).first()  
 **if** user **is None**:  
 **return False** g.current\_user = user  
 **return** user.check\_password(password)

Метод перевіряє чи є в системі такий користувач з заданим токеном, авторизує користувача, якщо такий був знайдений.  
@token\_auth.verify\_token  
**def** verify\_token(token):  
 g.current\_user = User.check\_token(token) **if** token **else None  
 return** g.current\_user **is not None**

Метод перевіряє чи є в системі такий девайс з заданим токеном, авторизує девайс, якщо такий був знайдений.@device\_token\_auth.verify\_token  
**def** verify\_device\_token(token):  
 device = TrackingDevice.check\_token(token) **if** token **else None** g.current\_device = device  
 **return** device **is not None**

Метод перевіряє чи авторизований користувач та чи відповідає користувач вказаним ролям, якщо все вірно, то виконується запит, який був вказаний, якщо ні, повертається помилка «Нема доступу».  
**def** roled\_login\_required(roles=[]):  
 **def** wrapper(fn):  
 @wraps(fn)  
 **def** decorated\_view(\*args, \*\*kwargs):  
 **if** g.current\_user **is None or not** g.current\_user.is\_authenticated:  
 **return** g.current\_app.login\_manager.unauthorized()  
 urole = g.current\_user.role  
 **if** len(roles) == 0 **and** urole **not in** roles:  
 **return** g.current\_app.login\_manager.unauthorized()  
 **return** fn(\*args, \*\*kwargs)  
 **return** decorated\_view  
 **return** wrapper

Приклад валідації данних запиту. Функція аналізує дані, які були передані в форматі JSON. Повертає помилку, якщо вхідні данні не є корректними, чи продовжує запит з новими фільтрованними даними, якщо вони пройшли валідацію.

**def** required\_fields(fields, optional=[]):  
 **def** wrapper(fn):  
 @wraps(fn)  
 **def** decorated\_view(\*args, \*\*kwargs):  
 data = request.get\_json(silent=**True**) **or** {}  
 data = Validator.validate(data, fields, optional)  
 **if not** data:  
 **return** bad\_request(101, \_l(**'must include all required fields'**))  
 request.passed\_data = data  
 **return** fn(\*args, \*\*kwargs)  
 **return** decorated\_view  
 **return** wrapper

Попередня функція використовує метод валідації. Функція перевіряє, чи вказані усі необходимі поля данних в запиті та фільтрує данні, якщо була вказана зайва інформація.

**class** Validator:  
 @staticmethod  
 **def** validate(data, required\_fields, optional\_fields):  
 result = {}  
 **for** field **in** required\_fields:  
 **if** field **not in** data **or not** str(data[field]):  
 **return None** result[field] = data[field]  
 **for** field **in** optional\_fields:  
 **if** field **in** data **and** str(data[field]):  
 result[field] = data[field]  
 **else**:  
 result[field] = **None  
 return** result

Приклад запиту на додавання нового продукту. Метод потребує авторизації користувача через токен та користувач повинен відповідати ролі адміністратора чи менеджера. Необходимі поля для додавання нового продукту: name – назва продукту, organization\_id – ідентифікаційний номер організації, яка виготовила продукт, та product\_type\_id – тип продукту. Запит повертає нову створену сущність продукту.  
@bp.route(**'/add-product'**, methods=[**'POST'**])  
@token\_auth.login\_required  
@roled\_login\_required(roles=[**'manager'**, **'admin'**])  
@required\_fields([**'name'**, **'organization\_id'**, **'product\_type\_id'**])  
**def** add\_product():  
 response = jsonify({**'product'**: Product.add\_product(request.passed\_data).serialize()})  
 response.status\_code = 200  
 **return** response