Project Report

Final project -jiading li

Tiny chemical web

Project Description: This website is developed for a small chemical business which sale materiel and catalyst. The normal user have to login to do the add products to the cart. And root user can add, update, delete products. There already add some products into database. Note: those products price and some information is not real.

Database Design: use sqlite which is a local database, below how database create. There exists three tables: User, Products, Cart

```
 CREATE TABLE Cart (

            id INTEGER NOT NULL,

            product_id INTEGER NOT NULL,

            user_id INTEGER NOT NULL,

            quantity INTEGER NOT NULL,

            number INTEGER NOT NULL,

            PRIMARY KEY (id),

            FOREIGN KEY(product_id) REFERENCES products (id),

            FOREIGN KEY(user_id) REFERENCES user (id)

),

CREATE TABLE products (

            id INTEGER NOT NULL,

            name VARCHAR(100) NOT NULL,

            price FLOAT NOT NULL,

            description TEXT NOT NULL,

            number INTEGER NOT NULL,

            image VARCHAR(100),

            category VARCHAR(100) NOT NULL,

            deleted BOOLEAN DEFAULT (0) NOT NULL,
	PRIMARY KEY (id),
	CHECK (deleted IN (0, 1))
)CREATE TABLE user (
	id INTEGER NOT NULL,
	firstname VARCHAR(20) NOT NULL,
	lastname VARCHAR(20) NOT NULL,
	email VARCHAR(120) NOT NULL,
	password VARCHAR(60) NOT NULL,
	PRIMARY KEY (id),
	UNIQUE (email)
```

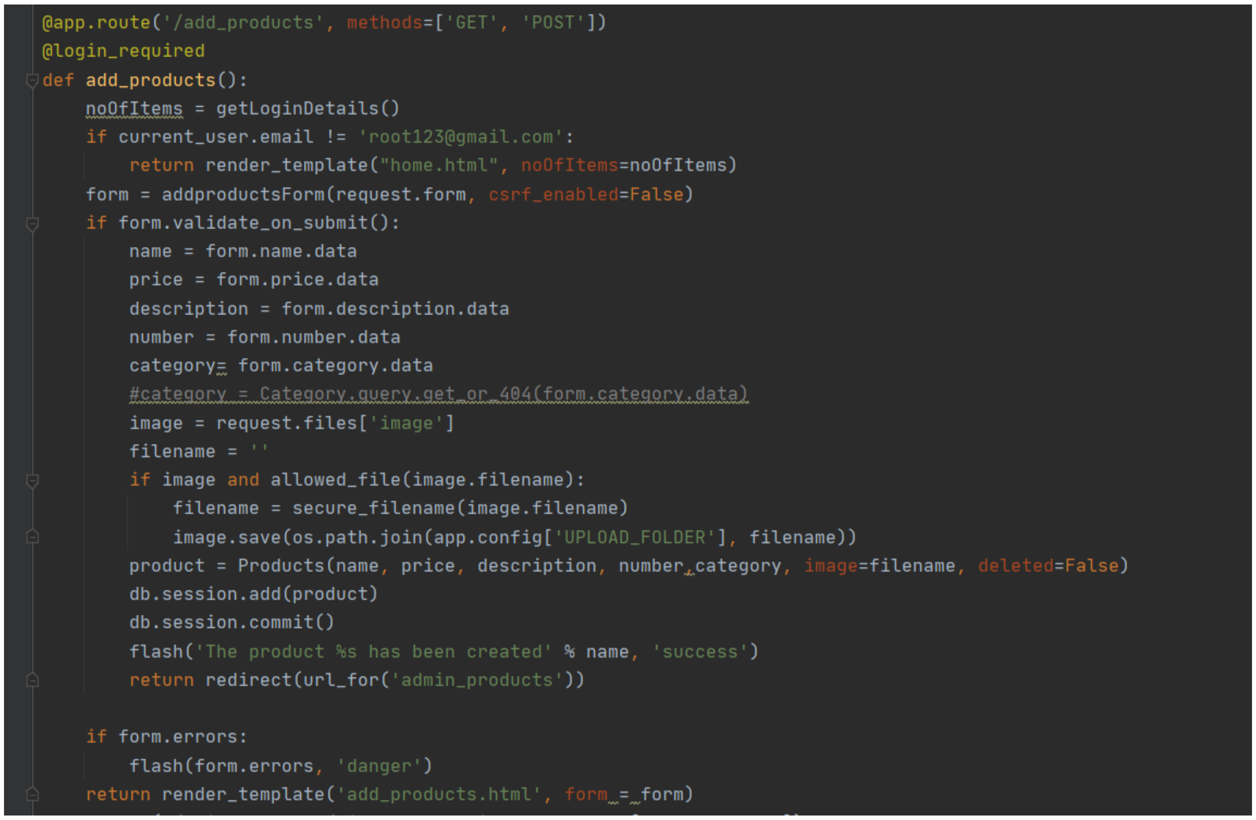
Languages/frameworks used for implementation: python-flask

Front-end: html/css/JavaScript

Back-end: flask+ mysql(sqlalchemy databse)

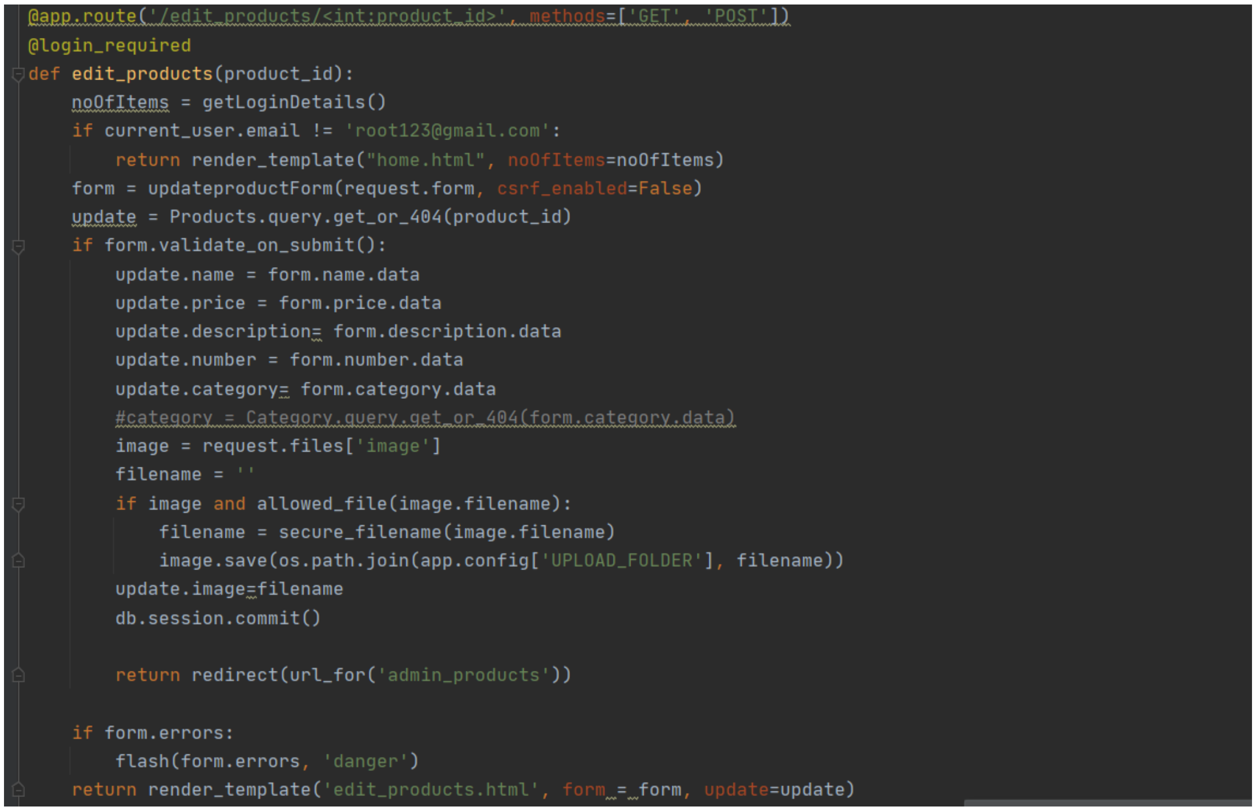
Main function:

Add-products:



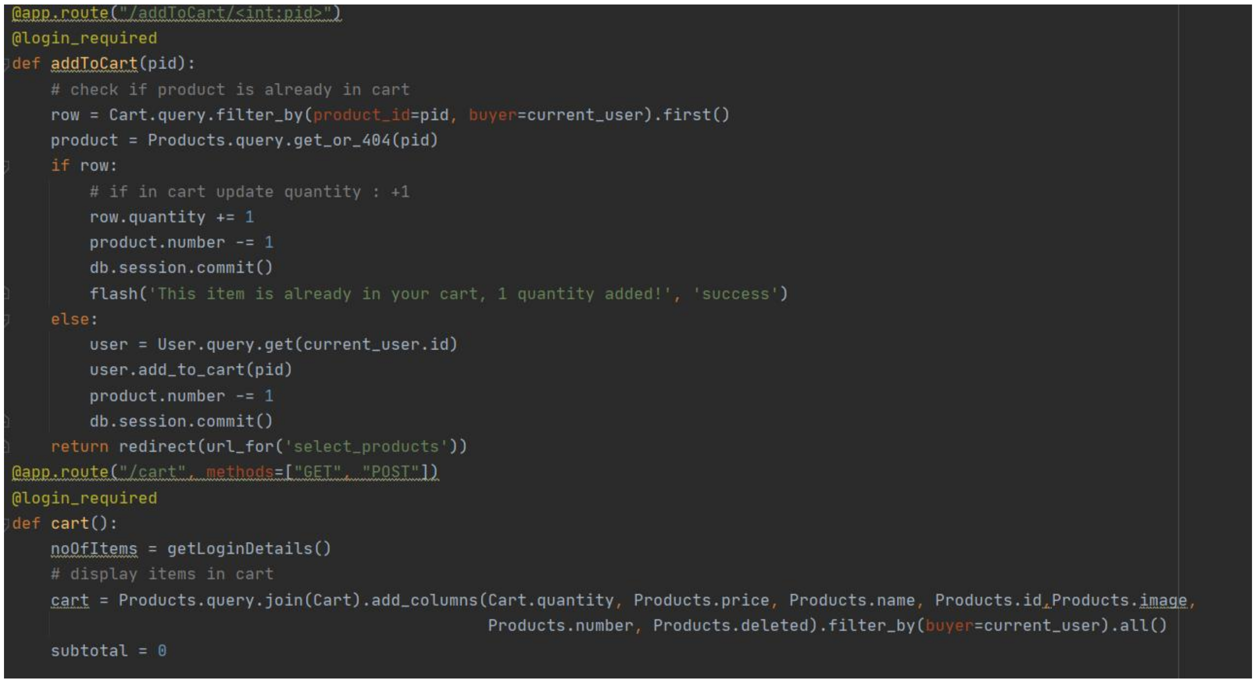
```
@app.route('/add_products', methods=['GET', 'POST'])
@login_required
def add_products():
    noOfItems = getLoginDetails()
    if current_user.email != 'root123@gmail.com':
        return render_template("home.html", noOfItems=noOfItems)
    form = addproductsForm(request.form, csrf_enabled=False)
    if form.validate_on_submit():
        name = form.name.data
        price = form.price.data
        description = form.description.data
        number = form.number.data
        category= form.category.data
        #category = Category.query.get_or_404(form.category.data)
        image = request.files['image']
        filename = ''
        if image and allowed_file(image.filename):
            filename = secure_filename(image.filename)
            image.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
        product = Products(name, price, description, number,category, image=filename, deleted=False)
        db.session.add(product)
        db.session.commit()
        flash('The product %s has been created' % name, 'success')
        return redirect(url_for('admin_products'))

    if form.errors:
        flash(form.errors, 'danger')
    return render_template('add_products.html', form = form)
```

Edit-product:

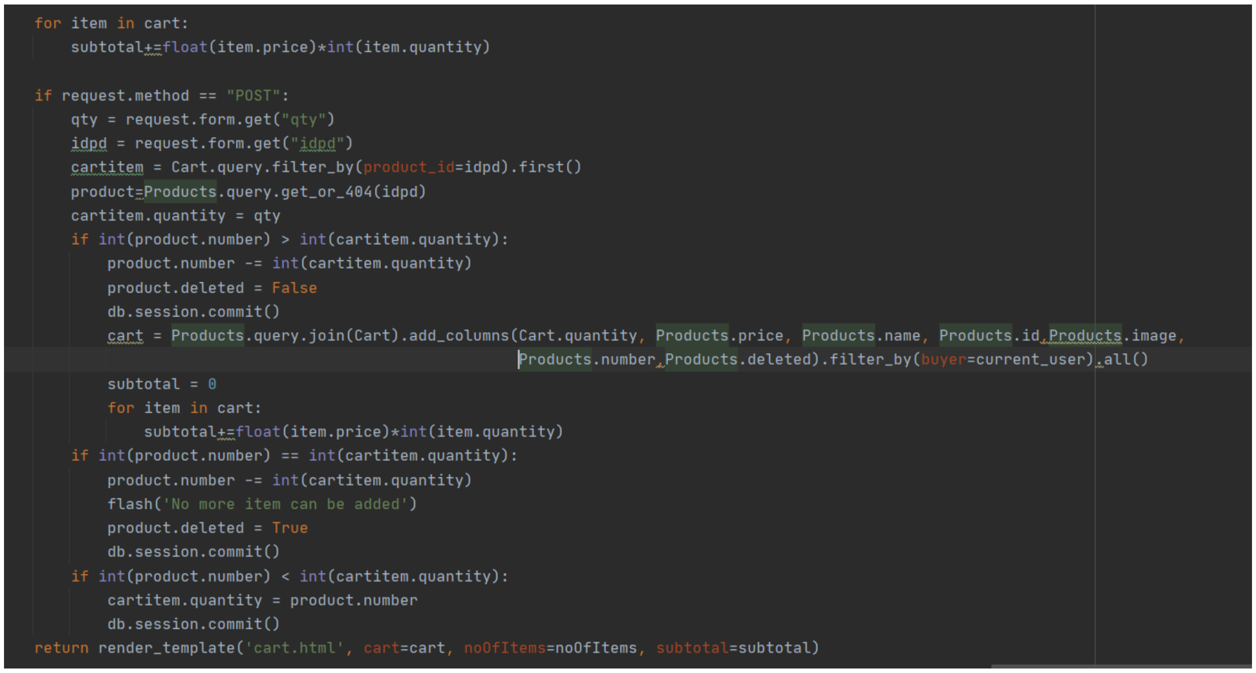


```python
@app.route('/edit_products/<int:product_id>', methods=['GET', 'POST'])
@login_required
def edit_products(product_id):
    noOfItems = getLoginDetails()
    if current_user.email != 'root123@gmail.com':
        return render_template("home.html", noOfItems=noOfItems)
    form = updateproductForm(request.form, csrf_enabled=False)
    update = Products.query.get_or_404(product_id)
    if form.validate_on_submit():
        update.name = form.name.data
        update.price = form.price.data
        update.description= form.description.data
        update.number = form.number.data
        update.category= form.category.data
        #category = Category.query.get_or_404(form.category.data)
        image = request.files['image']
        filename = ''
        if image and allowed_file(image.filename):
            filename = secure_filename(image.filename)
            image.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
        update.image=filename
        db.session.commit()

        return redirect(url_for('admin_products'))

    if form.errors:
        flash(form.errors, 'danger')
    return render_template('edit_products.html', form = form, update=update)
```

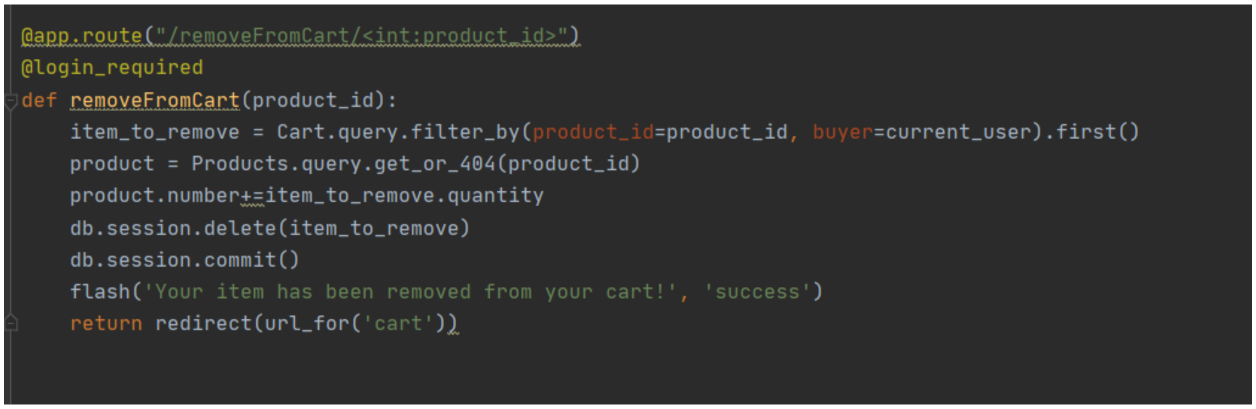
Addtocart & update &delete:



```python
@app.route("/addToCart/<int:pid>")
@login_required
def addToCart(pid):
    # check if product is already in cart
    row = Cart.query.filter_by(product_id=pid, buyer=current_user).first()
    product = Products.query.get_or_404(pid)
    if row:
        # if in cart update quantity : +1
        row.quantity += 1
        product.number -= 1
        db.session.commit()
        flash('This item is already in your cart, 1 quantity added!', 'success')
    else:
        user = User.query.get(current_user.id)
        user.add_to_cart(pid)
        product.number -= 1
        db.session.commit()
    return redirect(url_for('select_products'))
@app.route("/cart", methods=["GET", "POST"])
@login_required
def cart():
    noOfItems = getLoginDetails()
    # display items in cart
    cart = Products.query.join(Cart).add_columns(Cart.quantity, Products.price, Products.name, Products.id,Products.image,
                                                 Products.number, Products.deleted).filter_by(buyer=current_user).all()
    subtotal = 0
```

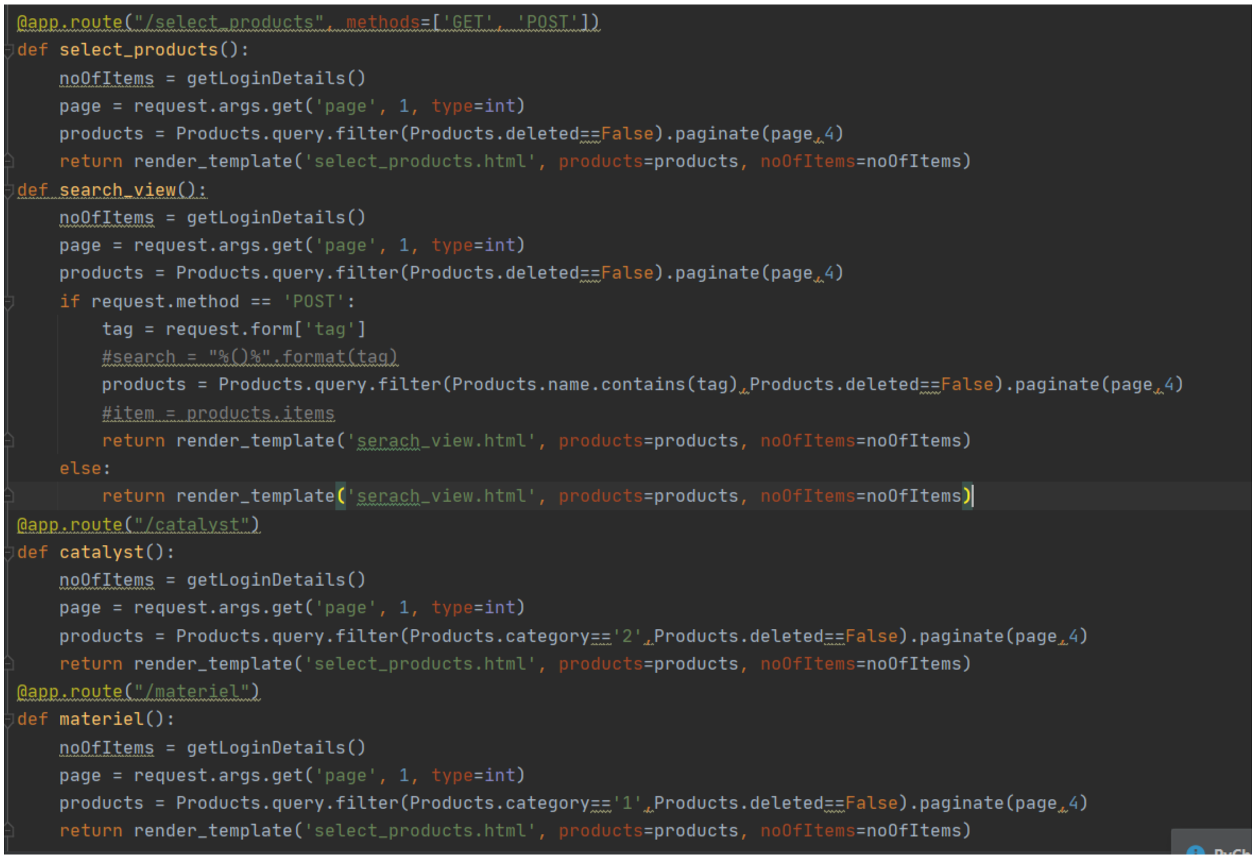


```python
    for item in cart:
        subtotal+=float(item.price)*int(item.quantity)

    if request.method == "POST":
        qty = request.form.get("qty")
        idpd = request.form.get("idpd")
        cartitem = Cart.query.filter_by(product_id=idpd).first()
        product=Products.query.get_or_404(idpd)
        cartitem.quantity = qty
        if int(product.number) > int(cartitem.quantity):
            product.number -= int(cartitem.quantity)
            product.deleted = False
            db.session.commit()
            cart = Products.query.join(Cart).add_columns(Cart.quantity, Products.price, Products.name, Products.id,Products.image,
                                                        Products.number,Products.deleted).filter_by(buyer=current_user).all()
            subtotal = 0
            for item in cart:
                subtotal+=float(item.price)*int(item.quantity)
        if int(product.number) == int(cartitem.quantity):
            product.number -= int(cartitem.quantity)
            flash('No more item can be added')
            product.deleted = True
            db.session.commit()
        if int(product.number) < int(cartitem.quantity):
            cartitem.quantity = product.number
            db.session.commit()
    return render_template('cart.html', cart=cart, noOfItems=noOfItems, subtotal=subtotal)
```



```python
@app.route("/removeFromCart/<int:product_id>")
@login_required
def removeFromCart(product_id):
    item_to_remove = Cart.query.filter_by(product_id=product_id, buyer=current_user).first()
    product = Products.query.get_or_404(product_id)
    product.number+=item_to_remove.quantity
    db.session.delete(item_to_remove)
    db.session.commit()
    flash('Your item has been removed from your cart!', 'success')
    return redirect(url_for('cart'))
```

Pagination & search& filter:



```
@app.route("/select_products", methods=['GET', 'POST'])
def select_products():
    noOfItems = getLoginDetails()
    page = request.args.get('page', 1, type=int)
    products = Products.query.filter(Products.deleted==False).paginate(page,4)
    return render_template('select_products.html', products=products, noOfItems=noOfItems)
def search_view():
    noOfItems = getLoginDetails()
    page = request.args.get('page', 1, type=int)
    products = Products.query.filter(Products.deleted==False).paginate(page,4)
    if request.method == 'POST':
        tag = request.form['tag']
        #search = "%{}%".format(tag)
        products = Products.query.filter(Products.name.contains(tag),Products.deleted==False).paginate(page,4)
        #item = products.items
        return render_template('serach_view.html', products=products, noOfItems=noOfItems)
    else:
        return render_template('serach_view.html', products=products, noOfItems=noOfItems)
@app.route("/catalyst")
def catalyst():
    noOfItems = getLoginDetails()
    page = request.args.get('page', 1, type=int)
    products = Products.query.filter(Products.category=='2',Products.deleted==False).paginate(page,4)
    return render_template('select_products.html', products=products, noOfItems=noOfItems)
@app.route("/materiel")
def materiel():
    noOfItems = getLoginDetails()
    page = request.args.get('page', 1, type=int)
    products = Products.query.filter(Products.category=='1',Products.deleted==False).paginate(page,4)
    return render_template('select_products.html', products=products, noOfItems=noOfItems)
```

Team Members: jiading li – jxl200013

Video url: https://drive.google.com/file/d/1PWx6cKf5zgg50V9Mro8xMdzC4JnF0gen/view?usp=sharing