



# **PYTHON AND SQL INTRO**

## **Project Report**

### **Online Cosmetic Shop, "FBeauty"**

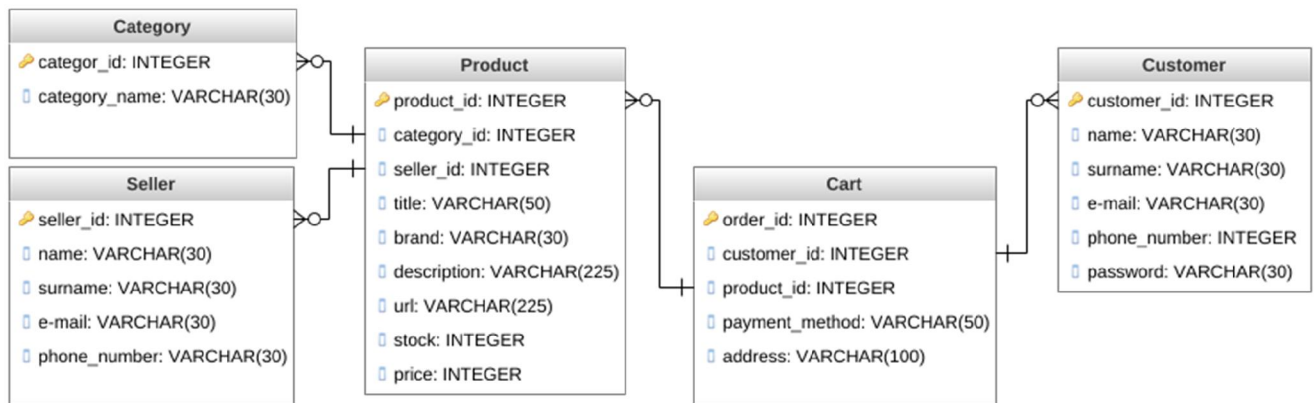
Nurdan Bešli, 457945

January, 2023

# Introduction

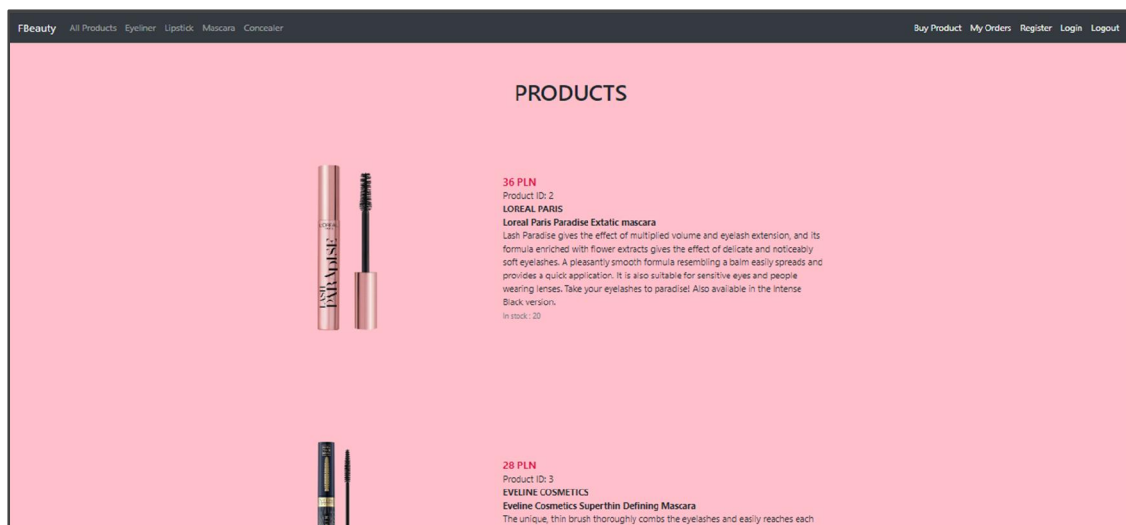
The web application, FBeauty, provides its users with the possibility to buy cosmetic products online. The user can see product details which are product image, price, title description and remaining stock. Users must register and login to create an order. After the login process, user can go to the buy product page, enter product id, payment method and delivery address. The user can buy one product at the same time, this means one order contains only one product. The user also can see orders created previously. There are verification steps in register, login and buy product processes and the error messages can be displayed in case of problem. The admin will be able to reach the databases, add and update information regarding sellers and their products.

## Database scheme



## Application Design

### Homepage & All Product Page



Users can see the product details on the Homepage or on All Products page. Product details include product image, price, title, brand, product id, description and remaining stocks. Only products whose stock is greater than zero are displayed. For example, the stock is zero for the product whose product id is 1 and category is mascara, so it is not displayed in both “All Product” and “Mascara” pages. Product id is located in these pages because the users should see and enter it in the “Buy Product” page to buy.

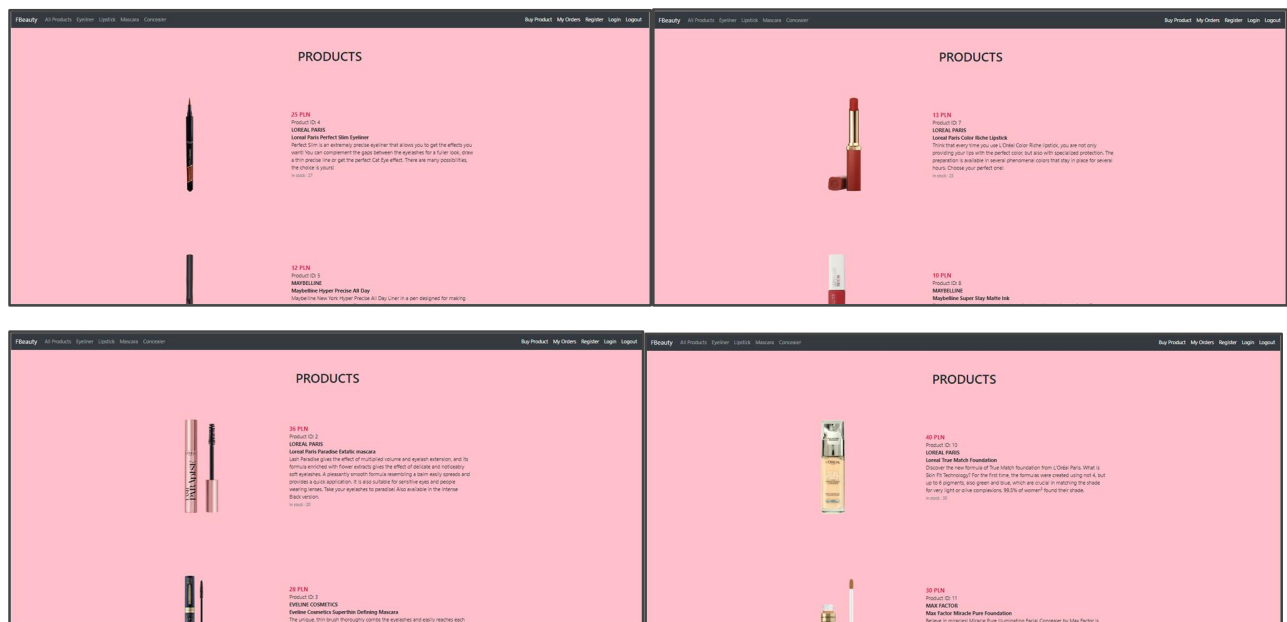
### SQL Query for Homepage & All Product Pages :

There is only one SQL query to show product details coming from product and category tables. After executing the query, it is assigned to the “products” variable and this variable sent to html codes. With the loop in html code, the product details are shown in the rows for each product in table attributes.

The code executed and assigned to “products” variable:

```
"SELECT product.product_id, product.category_id, product.title, product.brand, product.description, product.price, category.category_id, category.category_name, product.stock, product.url FROM product, category WHERE product.category_id = category.category_id and product.stock > 0"
```

### Eyeliner, Lipstick, Mascara and Concealer Pages



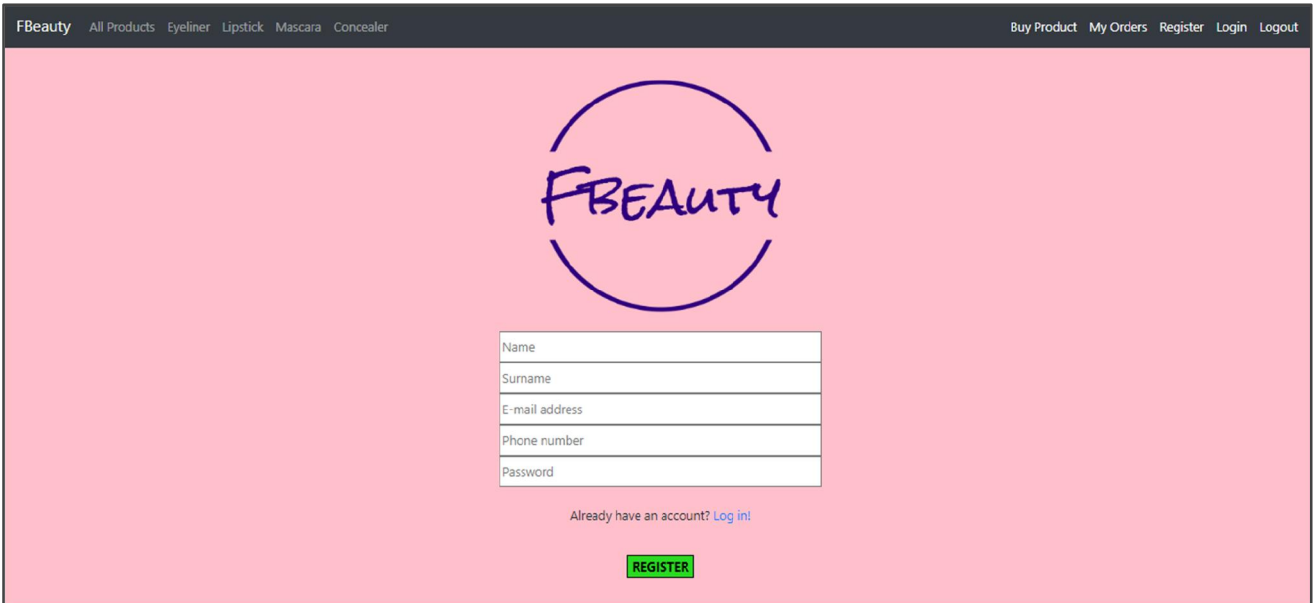
Products can also be seen as filtered in the category of products on the Eyeliner, Lipstick, Mascara and Concealer pages which can be reachable via navigation bar. In these pages, product image, price, product identity, brand, title, description and remaining stock information can be accessed.

### SQL Query for Eyeliner Page:

The SQL query is similar with the query of the homepage but an extra filter is added to show products as divided in categories. The below query is for the eyeliner page. For the other pages, "*product.category\_id = ?*" part is different. "?" is 2 for lipstick, 3 for mascara and 4 for concealer pages.

```
"SELECT product.product_id, product.category_id, product.title, product.brand, product.description, product.price, category.category_id, category.category_name, product.stock, product.url FROM product, category WHERE product.category_id = category.category_id and product.category_id = 1 and product.stock > 0"
```

### Register Page

The screenshot shows the 'FBeauty' Register Page. At the top, there is a dark navigation bar with links: 'FBeauty', 'All Products', 'Eyeliner', 'Lipstick', 'Mascara', 'Concealer', 'Buy Product', 'My Orders', 'Register', 'Login', and 'Logout'. The main content area has a light pink background. In the center, there is a large purple circular logo with the text 'FBEAUTY' inside. Below the logo is a registration form with five input fields: 'Name', 'Surname', 'E-mail address', 'Phone number', and 'Password'. Below the form, there is a link that says 'Already have an account? Log in!'. At the bottom of the form area, there is a green button labeled 'REGISTER'.

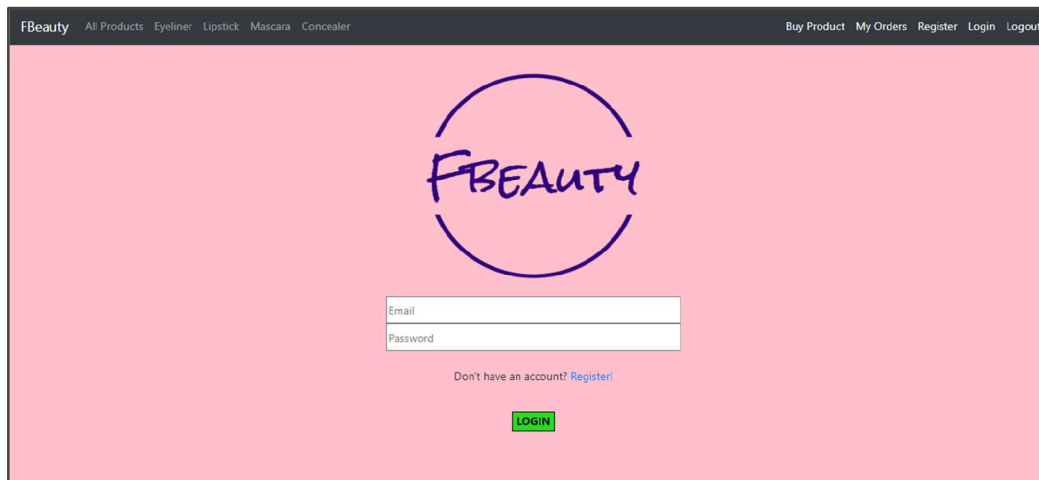
When users click on the Register button, they display the register page which contains name, surname, email, telephone and password form areas. All form areas on the register page must be filled. After clicking the Register button, the information entered by the user is recorded in the customer table of the database and the login page is displayed to the user.

### SQL Query for Register Page:

The information entered to name, surname, email, phone and password areas assigned to 5 variables created as a form in Flask and HTML. And these variables are used in the SQL code below to insert a record into the customer table.

```
""INSERT INTO customer (name,surname,email,phone,password) VALUES (?,?,?,?,"", (name,surname,email,phone,password))"
```

## Login Page



The screenshot shows the FBeauty login page. At the top, there is a dark navigation bar with the FBeauty logo on the left and links for 'Buy Product', 'My Orders', 'Register', 'Login', and 'Logout' on the right. The main content area has a light pink background. In the center, there is a large purple circular logo with the word 'FBEAUTY' in a stylized font. Below the logo, there are two input fields: 'Email' and 'Password'. Underneath these fields, there is a link that says 'Don't have an account? Register!'. At the bottom of the form, there is a green button labeled 'LOGIN'.

To view other pages, users must log in. If they click on Buy Product, My Orders, Login or Logout buttons in the navigation bar, they are forwarded to the Login page. On the Login page, users display the form with email and password areas. After all information is entered and clicking on the Login button, it is checked whether the information entered matches with a record in the customers table. If the information does not match, an error message is given and the information is requested to enter again. If the information matches, users are directed to the homepage.

### SQL Query for Login Page:

The information entered to email and password areas assigned to 2 variables created as a form in Flask and HTML. And these variables are used in the SQL code below to extract the record from the customer table. Other codes are used to check whether the entered information matched with any record or not.

```
""SELECT * FROM customer WHERE email=? AND password=?",(email,password)""
```

## Buy Product Page



The screenshot shows the FBeauty buy product page. It has the same dark navigation bar at the top as the login page. The main content area has a light pink background. In the center, there is a large purple circular logo with the word 'FBEAUTY' in a stylized font. Below the logo, there are two input fields: 'PRODUCT ID' and 'DELIVERY ADDRESS'. Underneath these fields, there is a dropdown menu for 'PAYMENT METHOD' with 'Credit Card' selected. At the bottom of the form, there is a green button labeled 'BUY PRODUCT'.

The user can go to the "Buy Product" page by clicking on the "Buy Product" Button in the navigation bar. Users should enter product ID, delivery address and choose one of the payment methods. After clicking the "Buy Product" button, the order is created and the Success page is displayed if product id is correct and the product stock is greater than zero. If the Product id is not in the Product table of the database or if the product stock is zero, the order is not created and the error message is displayed.

### **SQL Queries for Buy Product Page:**

The information entered to product id, delivery address and payment method areas assigned to 3 variables created as a form in Flask and HTML. And these variables are used in the SQL code below to insert a record into the cart table.

The below query checks whether there is a product whose product is equal to information entered in product id form area and also its stock is greater than zero or not. If there is no product with that product id or stock is zero, the result will be empty. And it helps us to show error messages on the page.

1. `"SELECT product_id FROM product WHERE product_id = " + product_id + " AND stock > 0"`

The below query is to extract the customer id of the user who has already logged in.

2. `"SELECT customer_id FROM customer WHERE email = " + session['email'] + """`

The below query is to insert a record into the cart table for creating a new order. In these query, customer\_id comes from 2nd query, product\_id comes from 1st query, payment method and delivery address come from the form areas.

3. `"INSERT INTO cart (customer_id, product_id, payment_method, address) VALUES (?, ?, ?, ?)",  
(customer_id, product, payment_method, address)"`

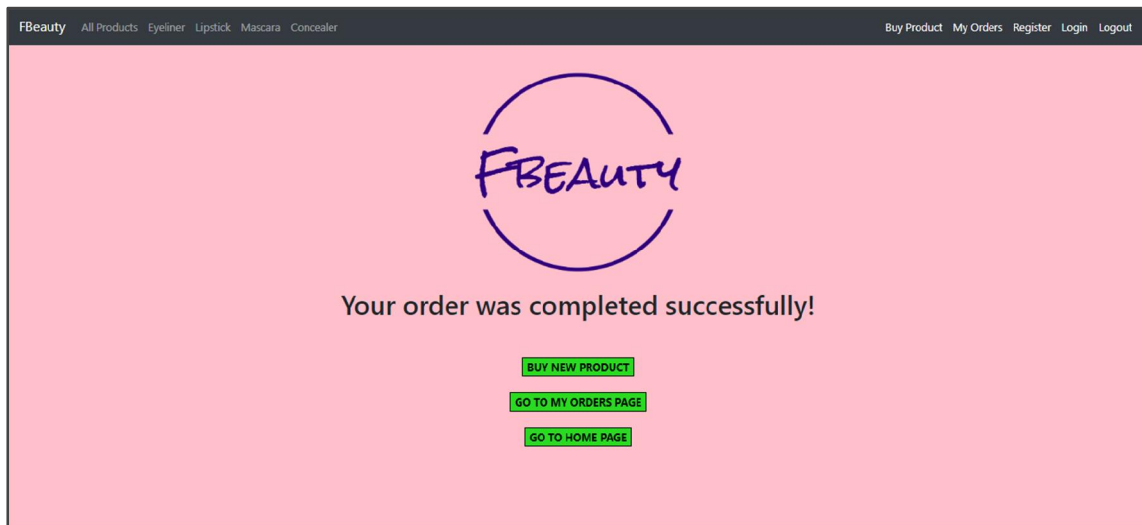
The below query is for extracting the current stock information of product whose product\_id entered in product id form area. This query is executed and its result assigned to the "stock" variable to use in the last query.

4. `"SELECT stock FROM product WHERE product_id = " + product_id + """`

The below query is for updating the stock information in the product table.

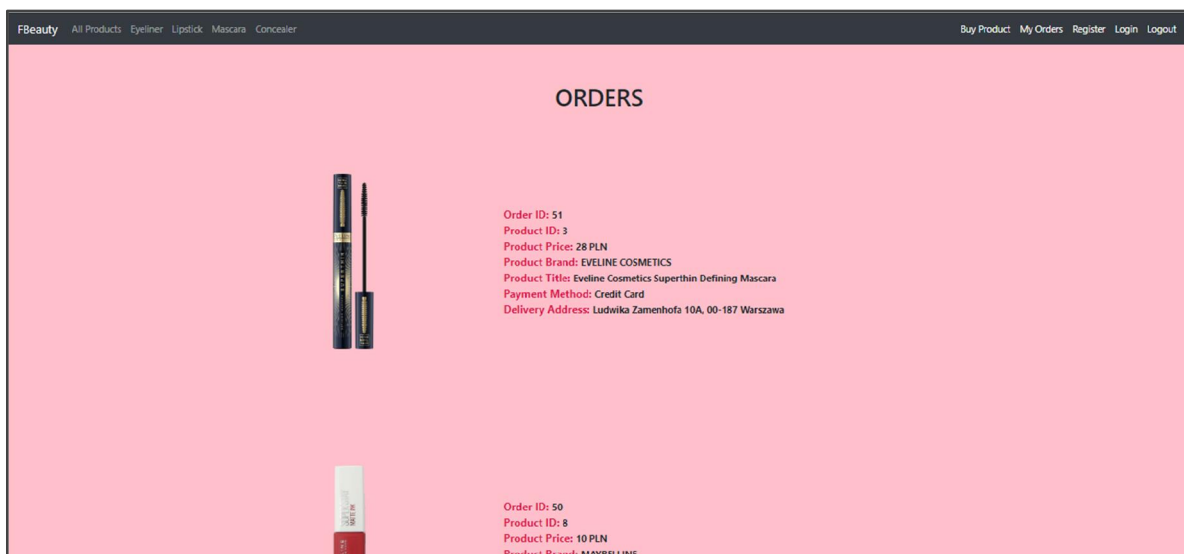
5. `"UPDATE product SET stock = {int(stock)-1} WHERE product_id = {product_id}"""`

## Success Page



The Success page contains the message that the order has been successfully created and 3 buttons that direct the user to "Buy Product", "My Orders" and "Home Page" pages.

## My Orders Page



In the "My Orders" page, the user can see orders previously created. Orders are shown starting from the latest created. Product information, payment method and delivery address information are displayed on this page.

### SQL Queries for My Orders Page:

The below query is used to extract the customer id of the user who has already logged in. This query is executed and assigned to the "customer\_id" variable to use in the 2nd query.

1. "SELECT customer\_id FROM customer WHERE email = '' + session['email'] + ''"

The SQL query below used to order and product information of the users logged in.

2. "SELECT product.product\_id, cart.order\_id, product.title, product.brand, product.price, product.url, cart.customer\_id, cart.payment\_method, cart.address FROM product, cart WHERE product.product\_id = cart.product\_id AND cart.customer\_id = {customer\_id} ORDER BY cart.order\_id DESC"

## Libraries

The packages imported can be seen below:

```
from flask import Flask, render_template, flash, redirect, url_for, session, request
import sqlite3
```