ShoppingDB

Plamen & Patryk

PseudoCode

Create category

```
Variable 1 = Increment category id
Variable2 = User inputs category name
Variable3 = User inputs category description
If category already exists
    Print already exists
Else
    Append with given variables
```

Create product

```
Variable 1 = Increment product id
Variable2 = User inputs product name
Variable3 = User inputs product price
Variable4 = User inputs product category
If category doesn't  exist
    Print doesn't exist
Else
    Append with given variables
```

PseudoCode

Create customer

```
Variable 1 = Increment customer id
Variable2 = User inputs customer email
Variable3 = User inputs customer phone number
Variable4 = User inputs customer address
Variables 5 and 6 = User inputs customer city and country
If email or phone number is invalid
        Print invalid and ask for a correct one
Else
        Append with given variables
```

Place order

```
Variable 1 = Increment order id
Variable2 = User inputs product id
Variable3 = User inputs the quantity
Variable4 = User inputs the customer id
Variables 5 and 6 = Calculated total price of the order and status set to shipping
Append with given variables
```

PseudoCode

Get sales by product id

```
Variable 1 = get list of ordered products by id

Variable 2 = get all occurrences of a product id in orders

Variable 3 = add quantities of all orders of the product

Variables 4 and 5= get product index to retrieve the name and price

Variable 6 = Calculated total price of the order and status set to shipping

Print product name, quantity sold and total sales
```

Get sales by category

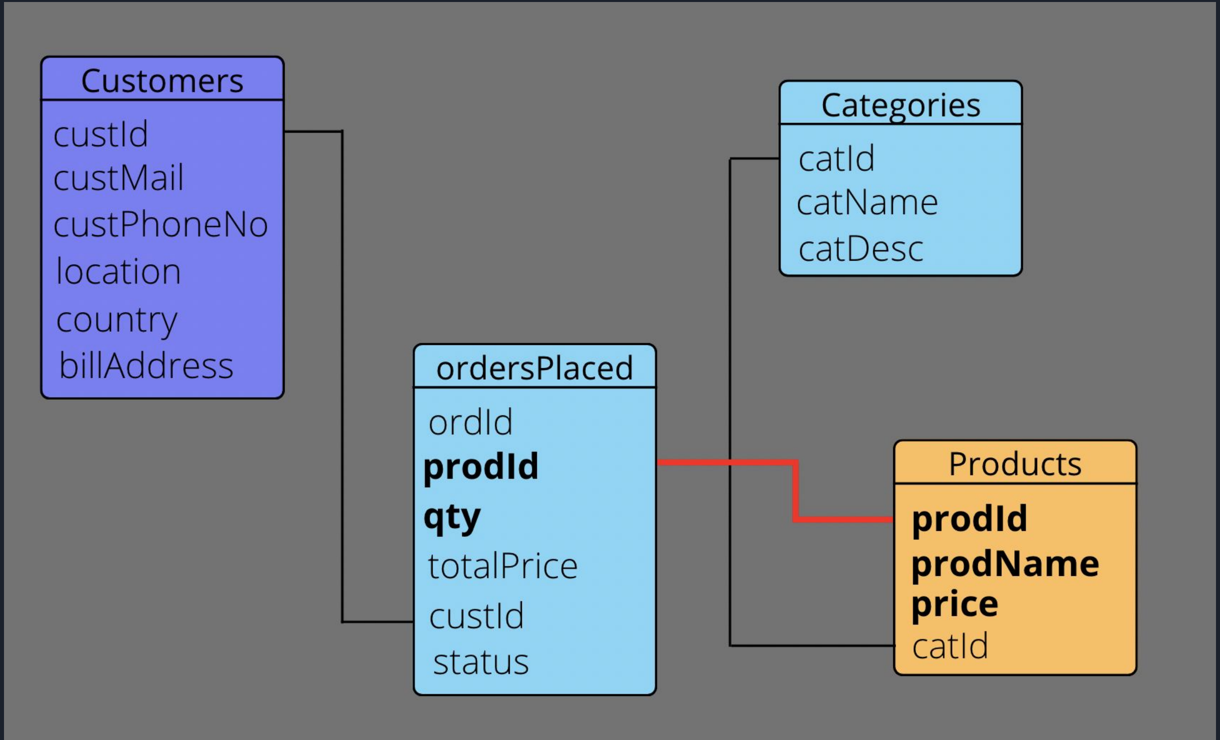
```
Variable 1 = get index of given category

Variable 2 = get the id of the given category

Variable 3 = get all occurrences of products of the category

For product in variable 3:

        Get sales by product id(product)
```

PseudoCode

Get sales ascending/descending

```
For product in products

Var1 = list of all products ordered

Var 2 = list of all occurrences of a  product

Var 3 = quantity of ordered products

Get index, name and price of the product

Push the product into a dictionary where key = product name and
val = total sales

Sort by value

Print dictionary
```

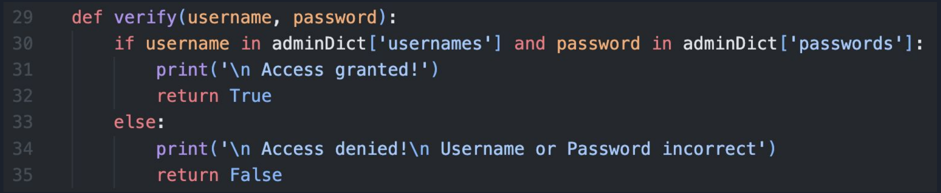
Flowchart



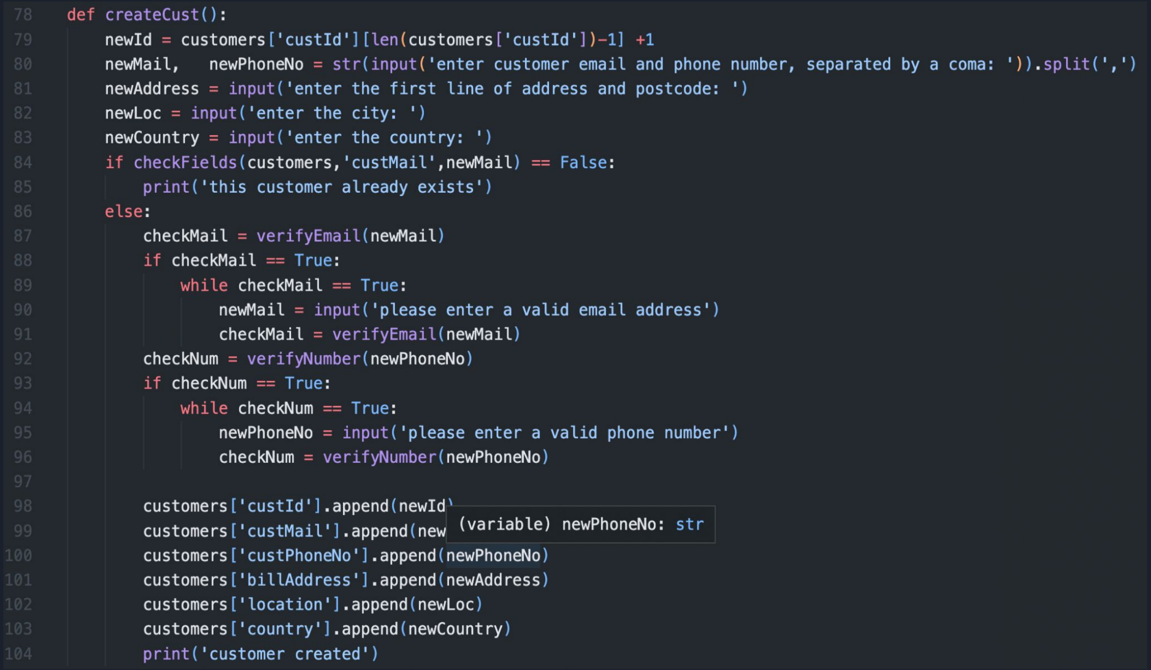
Get sales by ProductID:

ProdID -> ProdName -> price -> qty (**total price** = qty * price)

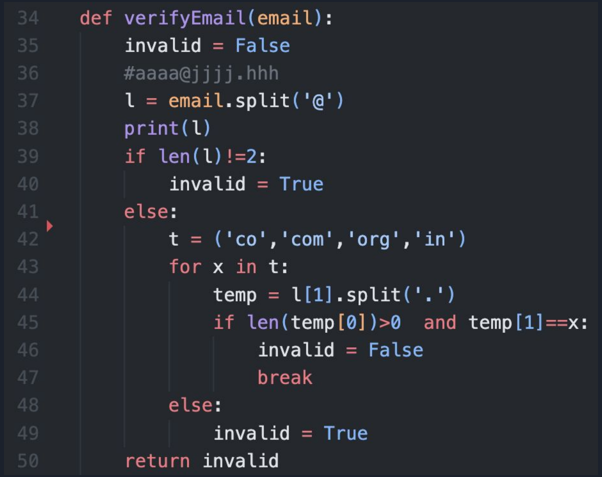
Code snippets



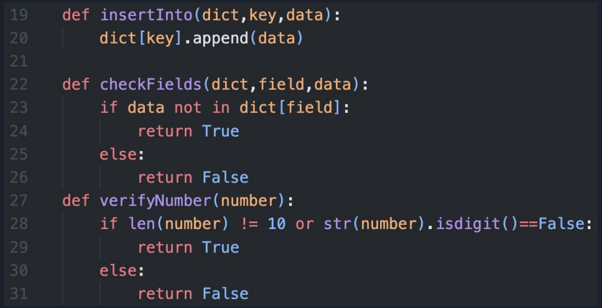
```python
def verify(username, password):
    if username in adminDict['usernames'] and password in adminDict['passwords']:
        print('\n Access granted!')
        return True
    else:
        print('\n Access denied!\n Username or Password incorrect')
        return False
```



```python
def createCust():
    newId = customers['custId'][len(customers['custId'])-1] +1
    newMail,   newPhoneNo = str(input('enter customer email and phone number, separated by a coma: ')).split(',')
    newAddress = input('enter the first line of address and postcode: ')
    newLoc = input('enter the city: ')
    newCountry = input('enter the country: ')
    if checkFields(customers,'custMail',newMail) == False:
        print('this customer already exists')
    else:
        checkMail = verifyEmail(newMail)
        if checkMail == True:
            while checkMail == True:
                newMail = input('please enter a valid email address')
                checkMail = verifyEmail(newMail)
        checkNum = verifyNumber(newPhoneNo)
        if checkNum == True:
            while checkNum == True:
                newPhoneNo = input('please enter a valid phone number')
                checkNum = verifyNumber(newPhoneNo)

        customers['custId'].append(newId
        customers['custMail'].append(new (variable) newPhoneNo: str
        customers['custPhoneNo'].append(newPhoneNo)
        customers['billAddress'].append(newAddress)
        customers['location'].append(newLoc)
        customers['country'].append(newCountry)
        print('customer created')
```

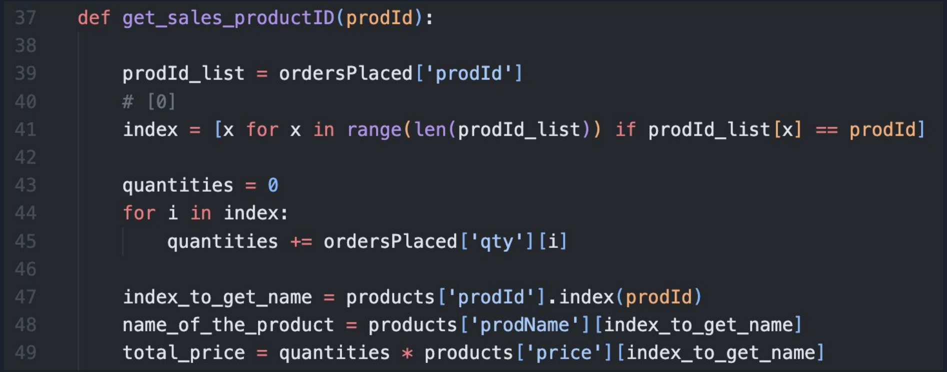


```python
def verifyEmail(email):
    invalid = False
    #aaaa@jjjj.hhh
    l = email.split('@')
    print(l)
    if len(l)!=2:
        invalid = True
    else:
        t = ('co','com','org','in')
        for x in t:
            temp = l[1].split('.')
            if len(temp[0])>0  and temp[1]==x:
                invalid = False
                break
        else:
            invalid = True
    return invalid
```

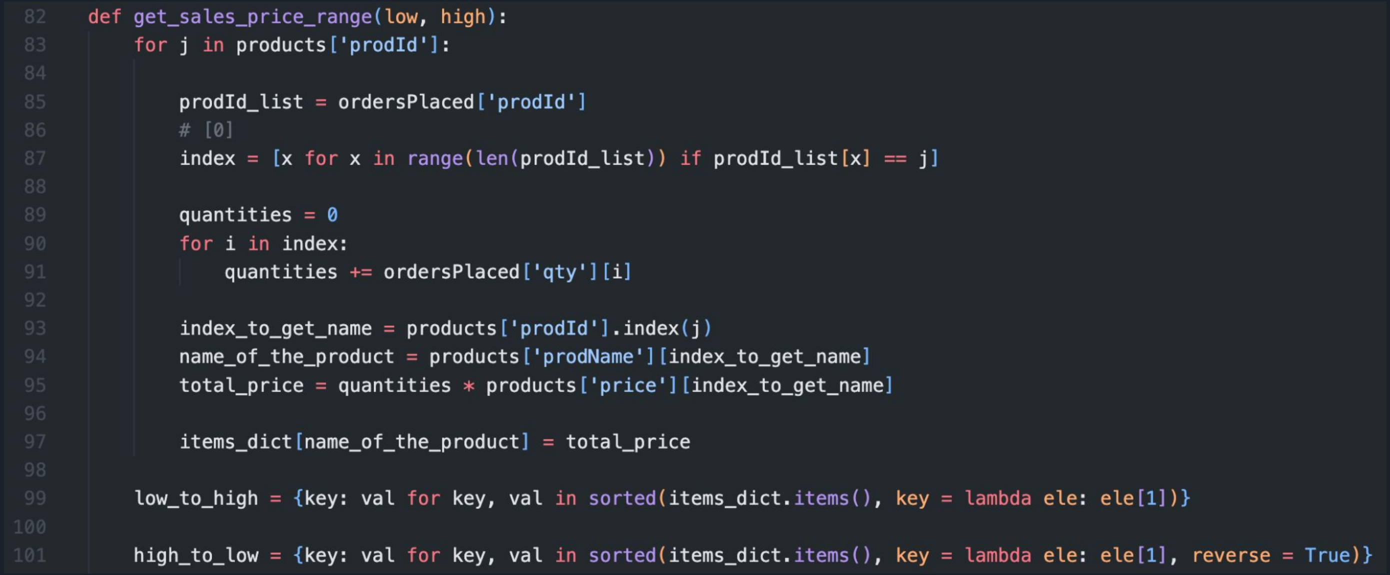


```python
def insertInto(dict,key,data):
    dict[key].append(data)

def checkFields(dict,field,data):
    if data not in dict[field]:
        return True
    else:
        return False
def verifyNumber(number):
    if len(number) != 10 or str(number).isdigit()==False:
        return True
    else:
        return False
```

Code snippets

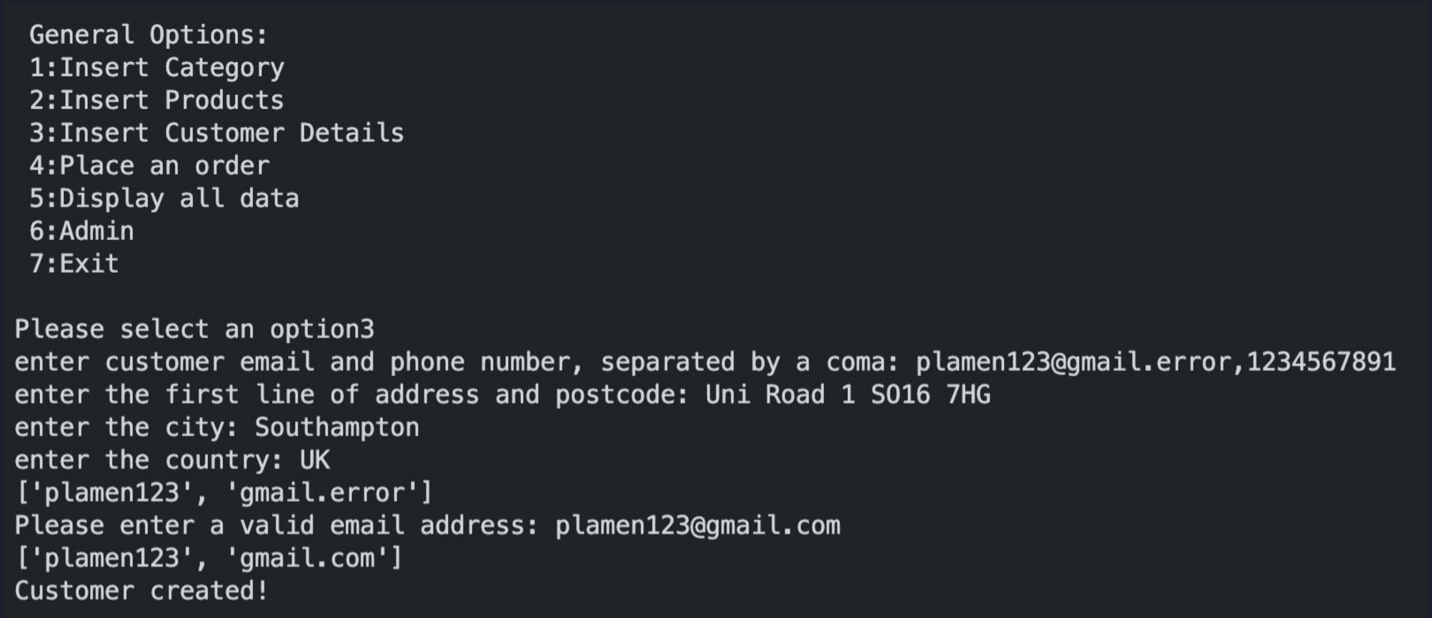


```python
def get_sales_productID(prodId):

    prodId_list = ordersPlaced['prodId']
    # [0]
    index = [x for x in range(len(prodId_list)) if prodId_list[x] == prodId]

    quantities = 0
    for i in index:
        quantities += ordersPlaced['qty'][i]

    index_to_get_name = products['prodId'].index(prodId)
    name_of_the_product = products['prodName'][index_to_get_name]
    total_price = quantities * products['price'][index_to_get_name]
```



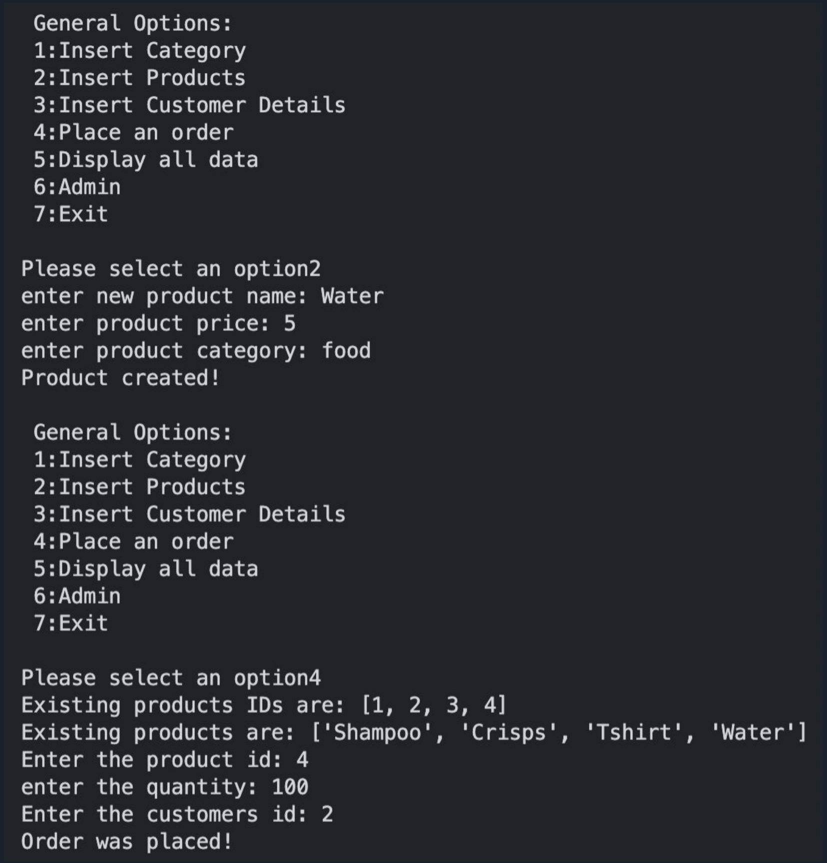
```python
def get_sales_price_range(low, high):
    for j in products['prodId']:

        prodId_list = ordersPlaced['prodId']
        # [0]
        index = [x for x in range(len(prodId_list)) if prodId_list[x] == j]

        quantities = 0
        for i in index:
            quantities += ordersPlaced['qty'][i]

        index_to_get_name = products['prodId'].index(j)
        name_of_the_product = products['prodName'][index_to_get_name]
        total_price = quantities * products['price'][index_to_get_name]

        items_dict[name_of_the_product] = total_price

    low_to_high = {key: val for key, val in sorted(items_dict.items(), key = lambda ele: ele[1])}

    high_to_low = {key: val for key, val in sorted(items_dict.items(), key = lambda ele: ele[1], reverse = True)}
```

Output

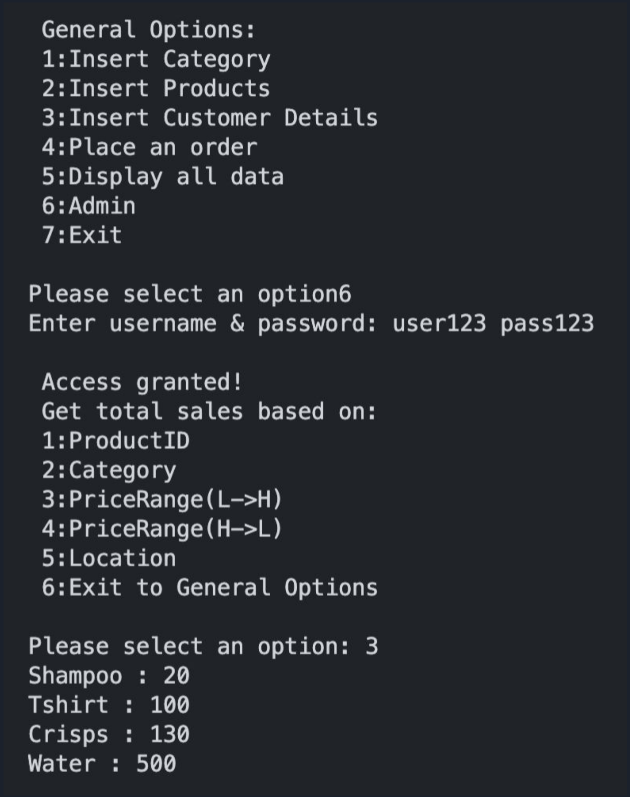


```
 General Options:
 1:Insert Category
 2:Insert Products
 3:Insert Customer Details
 4:Place an order
 5:Display all data
 6:Admin
 7:Exit

Please select an option3
enter customer email and phone number, separated by a coma: plamen123@gmail.error,1234567891
enter the first line of address and postcode: Uni Road 1 SO16 7HG
enter the city: Southampton
enter the country: UK
['plamen123', 'gmail.error']
Please enter a valid email address: plamen123@gmail.com
['plamen123', 'gmail.com']
Customer created!
```

Output



```
 General Options:
 1:Insert Category
 2:Insert Products
 3:Insert Customer Details
 4:Place an order
 5:Display all data
 6:Admin
 7:Exit

Please select an option2
enter new product name: Water
enter product price: 5
enter product category: food
Product created!

 General Options:
 1:Insert Category
 2:Insert Products
 3:Insert Customer Details
 4:Place an order
 5:Display all data
 6:Admin
 7:Exit

Please select an option4
Existing products IDs are: [1, 2, 3, 4]
Existing products are: ['Shampoo', 'Crisps', 'Tshirt', 'Water']
Enter the product id: 4
enter the quantity: 100
Enter the customers id: 2
Order was placed!
```



```
 General Options:
 1:Insert Category
 2:Insert Products
 3:Insert Customer Details
 4:Place an order
 5:Display all data
 6:Admin
 7:Exit

Please select an option6
Enter username & password: user123 pass123

 Access granted!
 Get total sales based on:
 1:ProductID
 2:Category
 3:PriceRange(L->H)
 4:PriceRange(H->L)
 5:Location
 6:Exit to General Options

Please select an option: 3
Shampoo : 20
Tshirt : 100
Crisps : 130
Water : 500
```