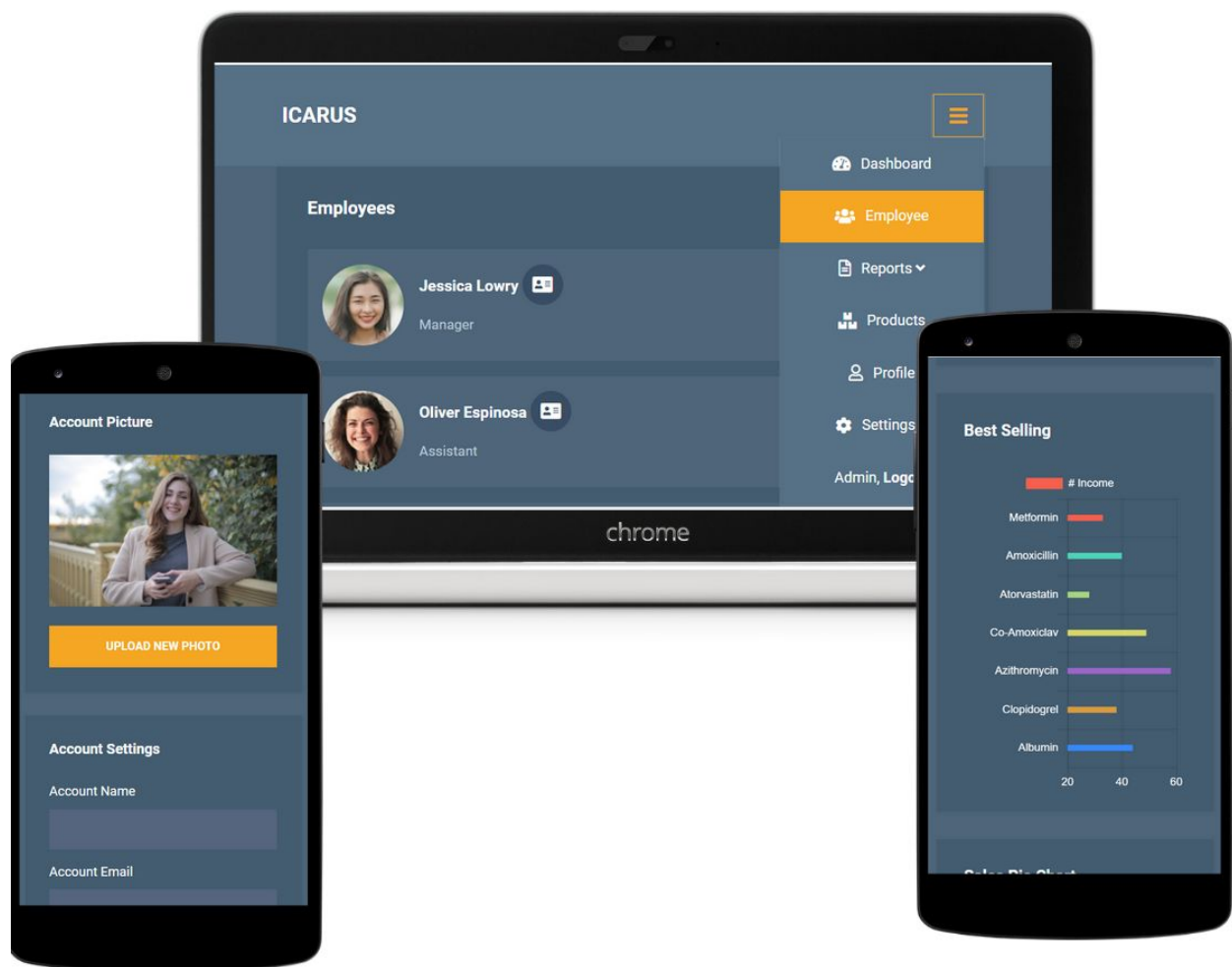


آزمایشگاه پایگاه داده

غزاله محمودی و یگانه مرشدزاده

۵ بهمن ماه ۱۳۹۹

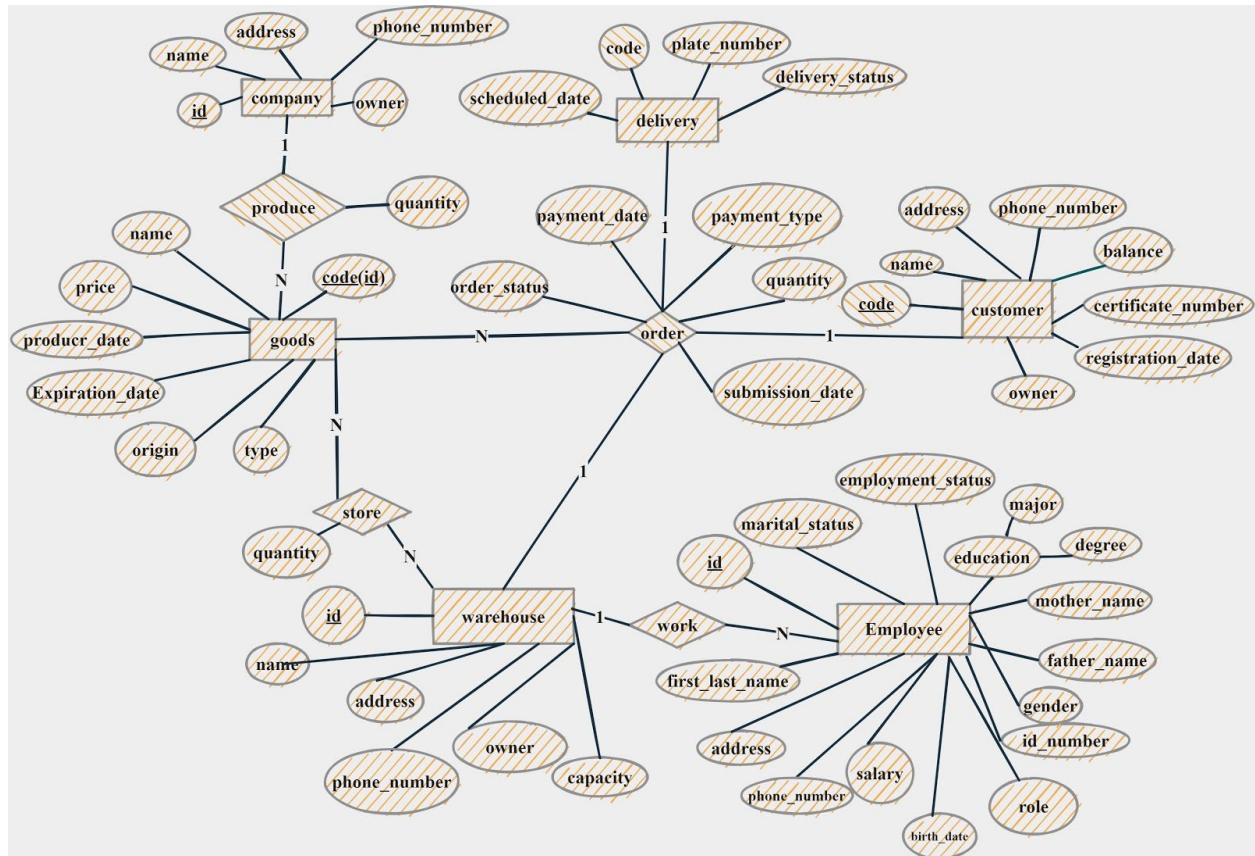
سیستم مدیریت فروش و انبارداری کالاهای دارویی



موجودیت ها

1. کالا (دارویی - آرایشی - بهداشتی - غذایی)
2. شرکت های دارویی (سازنده)
3. هر کالا از چه شرکتی خریداری شده
4. انبار
5. کالا های موجود در هر انبار (کد انبار - کد کالا - تعداد)
6. مشتری (داروخانه)
7. فروش (هر کالا به چه داروخانه ای فروخته شده و نوع خرید مشتری (نقدی - چکی - امانی))
8. تحویل سفارشات (در صورت درخواست مشتری توسط کارکنان بخش حمل نقل بسته تحویل داده میشود)
9. کارمندان

نمودار مدل ارتباط موجودیت



گزارش های دریافتی از سیستم

1. موجودی انبار به تفکیک نوع اقلام دارویی

○ `select * from store join goods on store.id_goods = goods.code_id`

■ به ازای یک کالای مشخص در کلیه انبار ها چه کالا هایی داریم.

○ `select * from store join goods on store.id_goods = goods.code_id join
warehouse on store.id_warehouse = warehouse.id`

■ به ازای یک انبار مشخص می خواهیم ببینیم از یک کالای مشخص چه مقدار داریم.

○ `select * from store join warehouse on store.id_warehouse = warehouse.id`

■ انبار هایی که در آن هایی جنس هایی ذخیره شده است.

2. موجودی انبار به تفکیک شرکت تولیدی

○ `select * from store join goods on store.id_goods = goods.code_id join
company on company.id = goods.company_id`

3. سفارشات هر داروخانه

○ `select * from order join on customer where order.id_customer = customer.code`

○ `select * from order join on customer where order.id_customer = customer.code join goods on order.id_goods = goods.code_id`

■ در اینجا علاوه بر اینکه سفارشات داروخانه خاصی را لیست می کند، جزئیات این کالا را که در جدول goods است نیز نشان میدهد. (مثلا ویژگی های استامینوفن را هم میآورد که داروخانه سفارش داده است)

4. گزارش تحویل هر سفارش (اینکه فلان سفارش خاص در چه وضعیتی است به مانند عدم پرداخت، ثبت نهایی، تحویل داده شده، در حال تحویل)

○ `select * from my_order join delivery on my_order.delivery_id = delivery.code join goods on my_order.id_goods = goods.code_id where order_status = 'PaidNDeli'`

■ گزارش تحویل هر سفارش بر اساس اینکه وضعیت آن سفارش چگونه است و جزئیات کالا قابل مشاهده است.

■ Status های مختلف نتایج مختلفی را به دنبال دارد.

5. سفارشات در لیست تحویل (وضعیت سفارش در حال تحویل است و برای ماشین حمل)

○ `select * from order join on delivery where order.delivery_id = delivery.code where delivery.status = "in progress"`

■ Status های مختلف نتایج مختلفی را به دنبال دارد.

6. سفارشات با وضعیت عدم پرداخت و نهایی شدن سفارش order report

- `select * from order join on customer where order.id_customer = customer.code where order_status = str`
 - `Str = "not paid" || ...`

7. گزارش اینکه هر انباری چه سفارش هایی برای تحویل دارد.

- `select * from order join on store where order.id_goods = store.id_goods join on warehouse warehouse.id = store.id_warehouse`

8. گزارش کارمند های شاغل در هر انبار

- `select * from employee join on warehouse where employee.warehouse_id = warehouse.id`

9. گزارش داروهای در شرف انقضا جهت ارائه تخفیف به منظور فروش رفتن قبل از منقضی شدن با `threshold = 10`

- `select * from goods where (Expiration_date - (SELECT CURRENT_DATE)) < 10`

10. گزارش فروش انواع کالا ها (اعم از بهداشتی، دارویی) به منظور اینکه ببینیم فروش کدام نوع جنس بهتر است.

- `select name , type , count(*) from goods join my_order on goods.code_id = my_order.id_goods group by goods.type order by count(*) DESC`

11. کالاهای موجود در سفارشات و تعداد آن ها که به صورت نزولی مرتب شدند.

- `select name , count(*) from goods join my_order on goods.code_id = my_order.id_goods group by goods.name order by count(*) DESC`

12. کالاهای موجود در انبار و تعداد آن ها که به صورت نزولی مرتب شدند.

- `select name , count(*) from goods join store on goods.code_id = store.id_goods group by goods.name order by count(*) DESC`

13. کالا موجود در انبار و تعداد آن ها که بر اساس نام و به صورت نزولی مرتب شدند.

- `select name , count(*) from goods group by goods.name order by count(*) DESC`

14. کالا موجود در انبار و تعداد آن ها که بر اساس کشور سازنده و به صورت نزولی مرتب شدند.

- select origin , count(*) from goods group by goods.origin order by coun .
- Function - Trigger

```
CREATE OR REPLACE FUNCTION check_price() RETURNS trigger AS $rval$
BEGIN
    IF NEW.price > 10000000 then
        update goods set price = 10000000 where code_id = NEW.code_id;
    END IF;
    RETURN NEW;
END;
$rval$ LANGUAGE 'plpgsql';

create trigger price after insert or update on goods
for each row
EXECUTE PROCEDURE check_price();
```

15. یک Procedure که به کمک آن می توان پیامی را نمایش داد.

- Procedure

```
CREATE OR REPLACE PROCEDURE display_message (INOUT msg TEXT)
AS $$
BEGIN
    RAISE NOTICE 'Procedure Parameter: %', msg ;
END ;
$$
LANGUAGE plpgsql ;

call display_message('This is my test case');
```

16. این کوئری مد نظر ما را بررسی می کند.

- Cursor

```
BEGIN;
DECLARE my_cursor CURSOR WITH HOLD FOR SELECT * FROM goods ORDER BY code_id;
COMMIT;

fetch 111 from my_cursor;
```


شرح پروژه

در این پروژه قصد داریم سامانه ای برای مدیریت شرکت های توزیع کننده دارو پیاده سازی کنیم.

وظیفه این شرکت ها به این صورت است که کالا ها را از شرکت های تولید کننده دارو، اقلام آرایشی و بهداشتی به صورت عمده خریداری کرده و با توجه به نیاز و سفارشات داروخانه ها، فروشگاه های اقلام بهداشتی و .. بین آنها توزیع میکنند.

بدین ترتیب هر داروخانه یا مرکز فروش جزئی کالا به جای بستن قرارداد با تعداد زیادی شرکت برای تامین کالاهای مورد نیاز خود همه سفارشات خود را به شرکت توزیع کالا می دهد و وظیفه قرارداد بستن با شرکت های تولید کننده اصلی به عهده شرکت توزیع قرار می گیرد. این کار به قرارداد های ما بین شرکت ها نظم می بخشد و موجب راحتی کار طرفین قرارداد می شود.

با توجه به دلایل مطرح شده وجود چنین شرکت هایی ضرورت می باید.

از طرفی برای مدیریت بهتر این شرکت ها ضروری است سامانه ای یکپارچه وجود داشته باشد. در این پروژه ما قصد داریم چنین سامانه ای را طراحی کنیم که مدیریت ، عملکرد و نظارت بر این شرکت ها را آسان می کند.

شرح موجودیت ها

کالا

منظور از کالا در این سیستم انواع اقلام دارویی، آرایشی و بهداشتی و غذایی است. این اقلام از کلیه ی شرکت های طرف قرارداد خریداری شده و در انبار ها نگهداری میشوند. این سیستم قابلیت مدیریت اجناس انبار را دارد. کارمندانی برای هر انبار تعریف شده اند که سیستم به آن ها اجازه مدیریت کالا ها (اضافه کردن، فروش) را دارند.

برای موجودیت کالا ویژگی هایی تعریف شده که عبارتند از :

- نام
- قیمت
- تاریخ تولید
- تاریخ انقضا
- شرکت تولید کننده
- نوع کالا
- هر کالا در کدام انبارها نگهداری میشود
- موجودی هر کالا در هر انبار

برای هر کالا شرکتی که از آن خریداری شده هم ثبت میشود.

شرکت های دارویی

این شرکت ها تولید کننده های اصلی دارویی، آرایشی و بهداشتی هستند. که انواع محصولات را تولید کرده و برای توزیع ان ها، ان ها را به شرکت های توزیع کننده میسپارند.

برای موجودیت کالا ویژگی هایی تعریف شده که عبارتند از :

- نام
- آدرس
- شماره تلفن
- مسئول فروش (شخص حقیقی طرف قرارداد به عنوان نماینده شرکت دارویی)
- قرار داد

برای هر انبار کارمندانی که در آن کار میکنند و مجوز تغییر کالاهای موجود در آن را دارند.

برای هر انبار موجودی کالا ها و اینکه از هر کالا به چه تعداد در این انبار موجود میباشد. به طور دقیق و با جزئیات ثبت میشود.

همچنین به طور دقیق شرکت تولید کننده هر کالا مشخص است.

انبار

هسته‌ی اصلی این شرکت ها انبار هایی هستند که کالا های خریداری شده از شرکت های دارویی (به طور مثال عبیدی) در آن ها نگهداری می شود. برای هر انبار، نام انبار، آدرس، شماره تلفن و گرداننده و مدیر انبار به عنوان اطلاعات اصلی در سیستم نگهداری میشود.

برای موجودیت کالا ویژگی هایی تعریف شده که عبارتند از :

- نام
- شماره تلفن
- آدرس
- مسئول انبار
- کالا های موجود در انبار
- تعداد موجود از هر کالا

کارمند

برای هر انبار کارمندانی که مجاز به تغییر در لیست کالاهای انبار هستند تعریف میشود. برای هر کارمند اطلاعاتی به مانند زیر تعریف می شود:

- نام و نام خانوادگی
- تاریخ تولد
- کد ملی
- شماره شناسنامه
- وضعیت تاهل
- شماره تماس
- آدرس
- جنسیت
- محل صدور
- نام پدر
- نام مادر
- تحصیلات
- رشته تحصیلی
- وضعیت استخدامی (رسمی/قراردادی/پاره وقت)
- نقش و جایگاه شغلی
- انباری که مجاز به تغییر در آن هستند
- حقوق

مشتری

مشتری های شرکت های تولید کننده دارو، داروخانه های بیمارستان ها، درمانگاه ها، پلی کلینیک ها و کلیه داروخانه های سطح شهر هستند که به صورت عمده اقدام به خرید انواع کالا ها میکنند.

شرکت های توزیع کننده این امکان را برای آن ها فراهم میکند که برای شرکت های مختلف هستند را با یک ثبت سفارش خریداری کنند. آن ها می توانند از چند شرکت مختلف خریداری کنند.

در واقع این داروخانه ها می توانند به جای اینکه به تک تک شرکت های مختلف سفارش دهند.

خریدهایی که توسط مشتری ثبت میشود میتواند به صورت نقدی، چکی، امانی و نسیه باشد. همچنین تحویل با خود مشتری و یا شرکت توزیع کننده میتواند باشد.

موجودیت مشتری شامل اطلاعات زیر است:

- کد مشتری
- نام و نام خانوادگی مسئول داروخانه
- شماره تماس
- آدرس
- اعتبار
- تاریخ عضویت
- کد ملی
- شماره شناسنامه
- شماره مجوز داروخانه

فروش / سفارش

ثبت سفارش توسط داروخانه در این شرکت صورت میگیرد. با توجه به نوع خرید و نوع دریافت محصول از انبار یا انبارها اجناس یکجا شده و به مشتری تحویل داده میشود.

- کد کالاها
- کد داروخانه
- کد انبار
- تعداد کالا های سفارشی هر نوع
- تاریخ ثبت
- نوع پرداخت
- تاریخ پرداخت
- وضعیت سفارش

status -> not paid, paid and not delivered, paid and self delivery, paid and delivered

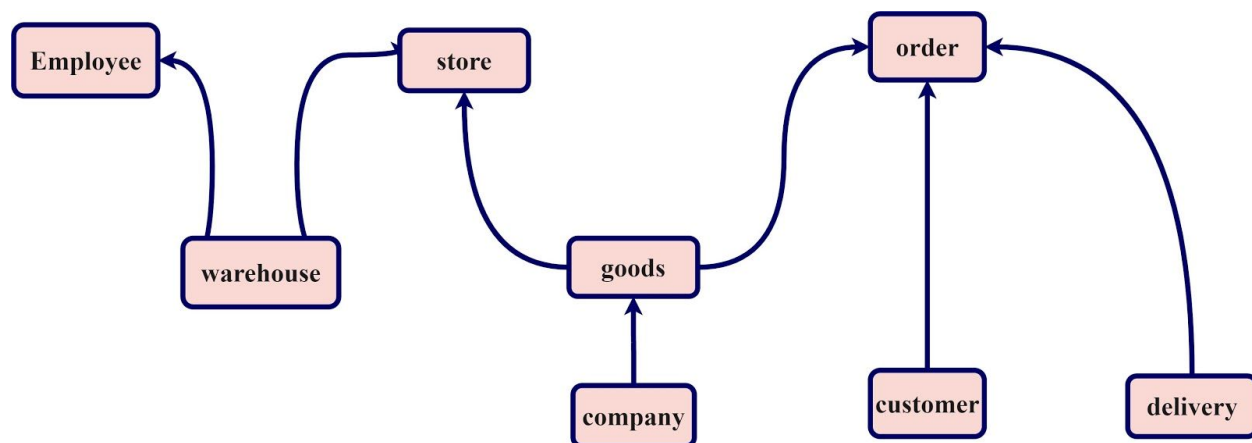
- نوع تحویل

تحويل و حمل و نقل سفارشات

پس از تسويه حساب سفارش، مشتری ميتواند انتخاب کند که کالا ها را خودش از درب مرکز فروش تحويل ميگيرد و يا توسط ماشين های حمل و نقل برايش ارسال شود. در صورتی که انتخاب کند که توسط مرکز به داروخانه تحويل داده شود، بايد اطلاعات زیر بدین منظور ثبت شوند:

- کد سفارش
- تاريخ برنامه ریزی شده برای تحويل
- وضعیت تحويل
- پلاک و کد ماشين حمل

گراف ارجاع



کد های پروژه

ساخت جداول در پایگاه داده

```
primary key(id_warehouse, id_goods));
create table delivery(
    code integer not null primary key ,
    delivery_status varchar(10) check (delivery_status in
('InProgress', 'Delivered', 'NotProc')),
    scheduled_date date,
    plate_number varchar(30));

create table my_order(
    id_goods integer not null,
    delivery_id integer not null,
    id_customer integer not null,
    submission_date date not null,
    order_status varchar(12) check (order_status in ('NPaid',
'PaidNDeli', 'PaidSelfDeli','PaidDeliD')),
    payment_type varchar(6) check (payment_type in ('online', 'cash',
'cheque')),
    Payment_date date,
    quantity integer,
    foreign key(id_goods) references goods(code_id),
    foreign key(delivery_id) references Delivery(code),
    foreign key(id_customer) references customer(code),
    primary key(id_goods, delivery_id, id_customer,
submission_date));

create table customer(
    code integer not null primary key ,
    name varchar(30) not null ,
```

```

owner varchar(30) not null ,
phone_number varchar(15),
address varchar(50),
balance integer,
registration_date date ,
certificate_number varchar(30));

create table employee(
  employee_id integer not null primary key ,
  first_name varchar(30) not null,
  last_name varchar(30),
  marital_status integer,
  father_name varchar(30),
  mother_name varchar(30),
  national_code varchar(10),
  education varchar(30),
  employment_status varchar(30) check (employment_status in
('Permanent ', 'Temporary', 'Independent')),
  phone_number varchar(15),
  address varchar(50),
  salary integer,
  birth_date date,
  gender bit,
  role varchar(30),
  warehouse_id integer,
  foreign key(warehouse_id) references warehouse(id));

```

درج اطلاعات در پایگاه داده

```
insert into company values (1, 'company1', 'poonak sqrt' , '0212345678',
'owner1');
insert into company values (2, 'company2', 'poonak2 sqrt' , '0222345678',
'owner2');

insert into warehouse values (11, 'warehouse11', 'tajrish sqrt' ,
'02111234567', 'owner11', 11);
insert into warehouse values (22, 'warehouse22', 'tajrish2 sqrt' ,
'02122234567', 'owner22', 22);

insert into goods values (111, 1, 'goods111', 111000, '2020-01-01',
'2021-01-01', 'Iran', 'Medi');
insert into goods values (222, 1, 'goods222', 222000, '2020-02-02',
'2022-02-02', 'Germany', 'Heal');
insert into goods values (333, 1, 'goods333', 333000, '2020-03-03',
'2023-03-03', 'France', 'Cosm');
insert into goods values (444, 2, 'goods444', 444000, '2020-04-04',
'2024-04-04', 'Turkey', 'Heal');
insert into goods values (555, 2, 'goods555', 555000, '2020-05-05',
'2025-05-05', 'Russia', 'Cosm');

update goods set origin = 'Iran' where goods.code_id = 222
/*be khatere foreign key delete nashod case case bayad ezafe kard
delete from goods where goods.code_id = 555*/

select * from goods

insert into store values (11, 111, 1111);
insert into store values (11, 333, 2222);
insert into store values (22, 444, 3333);

insert into customer values (1111, 'customer1111', 'ownercustomer1111',
'0211111345', 'vanak1 sqrt', 11110000, '2020-01-11', 111100001111);
insert into customer values (2222, 'customer2222', 'ownercustomer2222',
'02112222345', 'vanak2 sqrt', 22220000, '2020-02-22', 222200002222);

insert into delivery values (11111, 'InProgress', '2020-01-21', '1111100000');
insert into delivery values (22222, 'Delivered', '2020-02-22', '2222200000');
```

```

insert into delivery values (33333, 'NotProc', '2020-03-23', '3333300000');

insert into my_order values (333, 11111, 1111, '2020-11-25', 'PaidNDeli',
'online', '2020-11-15', 13);
insert into my_order values (444, 22222, 2222, '2020-11-25', 'PaidSelfDeli',
'cash', '2020-11-15', 13);
insert into my_order values (555, 33333, 2222, '2020-11-25', 'PaidDeliD',
'cheque', '2020-11-15', 13);

insert into employee values (1111111, 'First1111111', 'Last1111111',
'1','father1111111','mother1111111','1111111000',
'bachelor of management', 'Permanent ', '11111110', 'resalat sqrt', 1111111,
'1991-01-01', '0', 'manager', 11);
insert into employee values (2222222, 'First2222222', 'Last2222222',
'0','father2222222','mother2222222','2222222000',
'master of industrial', 'Temporary', '22222220', 'resalat2 sqrt', 2222222,
'1992-02-02', '1', 'manager', 22);

```

app.py

در این فایل کدهای مربوط به اتصال flask به html نوشته شده است.

```
from flask import Flask , render_template, request, redirect, url_for
import data as Database
import sys
import json

app = Flask(__name__)
user = []

@app.route('/home', methods=['POST', 'GET'])
def home():

    return render_template('index.html')

@app.route('/products', methods=['POST', 'GET'])
def products():
    products = Database.get_product()

    value= -1 #default value
    if request.method == "POST":
        value = request.form

        if value['value'] == '0' :
            products = Database.get_product()
        else:
            products = Database.get_product_by_category(value['value'])

    return render_template('products.html', products = products)

@app.route('/employee')
def employee():
    employee = Database.employee()
    return render_template('employee.html', employees=employee)

@app.route('/report-order', methods=['POST', 'GET'])
def report_order():
```



```

order = Database.get_order()

value= -1 #default value
if request.method == "POST":
    value = request.form
    if value['orderValue'] == '0' and value['deliveryValue'] == '0':
        order = Database.get_order()

    elif value['deliveryValue'] != '0' and value['orderValue'] == '0':
        order =
Database.get_order_filter_by_delivery(value['deliveryValue'])

    elif value['orderValue'] != '0' and value['deliveryValue'] == '0':
        order = Database.get_order_list(value['orderValue'])

    elif value['orderValue'] != '0' and value['deliveryValue'] != '0':
        order =
Database.get_order_list_and_filter_by_delivery(value['orderValue'],
value['deliveryValue'])
    else :
        order = Database.get_order()

print(order)
return render_template('report-order.html', report_orders = order)

@app.route('/report-sell')
def report_sell():
    return render_template('report-sell.html')

@app.route('/add-product', methods=['POST', 'GET'])
def add_product():

    if request.method == "POST":
        data = request.form
        print("data", data)

        if not isdigit(data['id']) or not isdigit(data['price']) or not
isdigit(data['quantity']):
            message = "Invalid input"
            print(message)
            render_template('add-product.html', message=message)
        else :
            Database.insert_product(data)

```

```

        return redirect(url_for('products'))

    return render_template('add-product.html')

@app.route('/edit-product')
def edit_product():
    return render_template('edit.html')

@app.route('/accounts')
def accounts():
    return render_template('accounts.html')

@app.route('/login', methods=['POST', 'GET'])
def login():
    message = ''

    if request.method == "POST":
        value = request.form
        print(value)
        if value['username'] == 'fleur_masson@outlook.com' and
value['password'] == 'wer1234' :
            message = "Login successfully!"
            print(message)
            return redirect(url_for('home'))
        else:
            message = "Email or pass incorrect! Please try again!"
            print(message)
            return render_template('login.html', message = message)

    return render_template('login.html')

if __name__ == "__main__":
    app.run()

```

data.py

در این فایل کدهای مربوط به اتصال flask به دیتابیس pgAdmin نوشته شده است.

```
import psycopg2
from psycopg2.extras import DictCursor
import sys

connection = psycopg2.connect(user      = "postgres",
                              password = "1234321A@m",
                              host      = "localhost",
                              port      = "5432",
                              database  = "Icarus")

cursor = connection.cursor(cursor_factory = DictCursor)

def get_product():
    cursor.execute("select * from store join goods on store.id_goods =
goods.code_id")
    record = cursor.fetchall()

    return record

def get_product_by_category(category):
    query = "select * from store join goods on store.id_goods = goods.code_id
where type = '"+category+"'"
    cursor.execute(query)
    record = cursor.fetchall()
    return record

def insert_product(data):
    print(data)
    print("insert_product")
    query = "insert into goods values
("+data['id']+",2,"'+data['name']+"'", "+data['price']+", "'"+data['produc
tion_date']+"'", "+data['Expiration_date']+"'", "+data['origin']+"'",
"+data['value']+"'"")

    print(query)
    cursor.execute(query)
    connection.commit()
```

```

        query_ = "insert into store values (11,"+data['id']+",""+
data['quantity']+")"
        print(query_)
        cursor.execute(query_)

        connection.commit()

def get_order():
    query = "select * from my_order join delivery on my_order.delivery_id =
delivery.code"
    cursor.execute(query)
    record = cursor.fetchall()
    return record

def get_order_list(order_status):
    query = "select * from my_order join delivery on my_order.delivery_id =
delivery.code where order_status = '"+order_status+"'"
    cursor.execute(query)
    record = cursor.fetchall()
    return record

def get_order_filter_by_delivery(delivery_status):
    query = "select * from my_order join delivery on my_order.delivery_id =
delivery.code where delivery_status = '"+delivery_status+"'"
    cursor.execute(query)
    record = cursor.fetchall()
    return record

def get_order_list_and_filter_by_delivery(order_status, delivery_status):
    query = "select * from my_order join delivery on my_order.delivery_id =
delivery.code where delivery_status = '"+delivery_status+"' and order_status
='"+order_status+"'"
    cursor.execute(query)
    record = cursor.fetchall()
    return record

def employee():
    query = "select * from employee"
    cursor.execute(query)
    record = cursor.fetchall()
    return record

```

اسکرین صفحات پروژه

Login Page

ICARUS

Welcome to Icarus

Email

fleur_masson@outlook.com

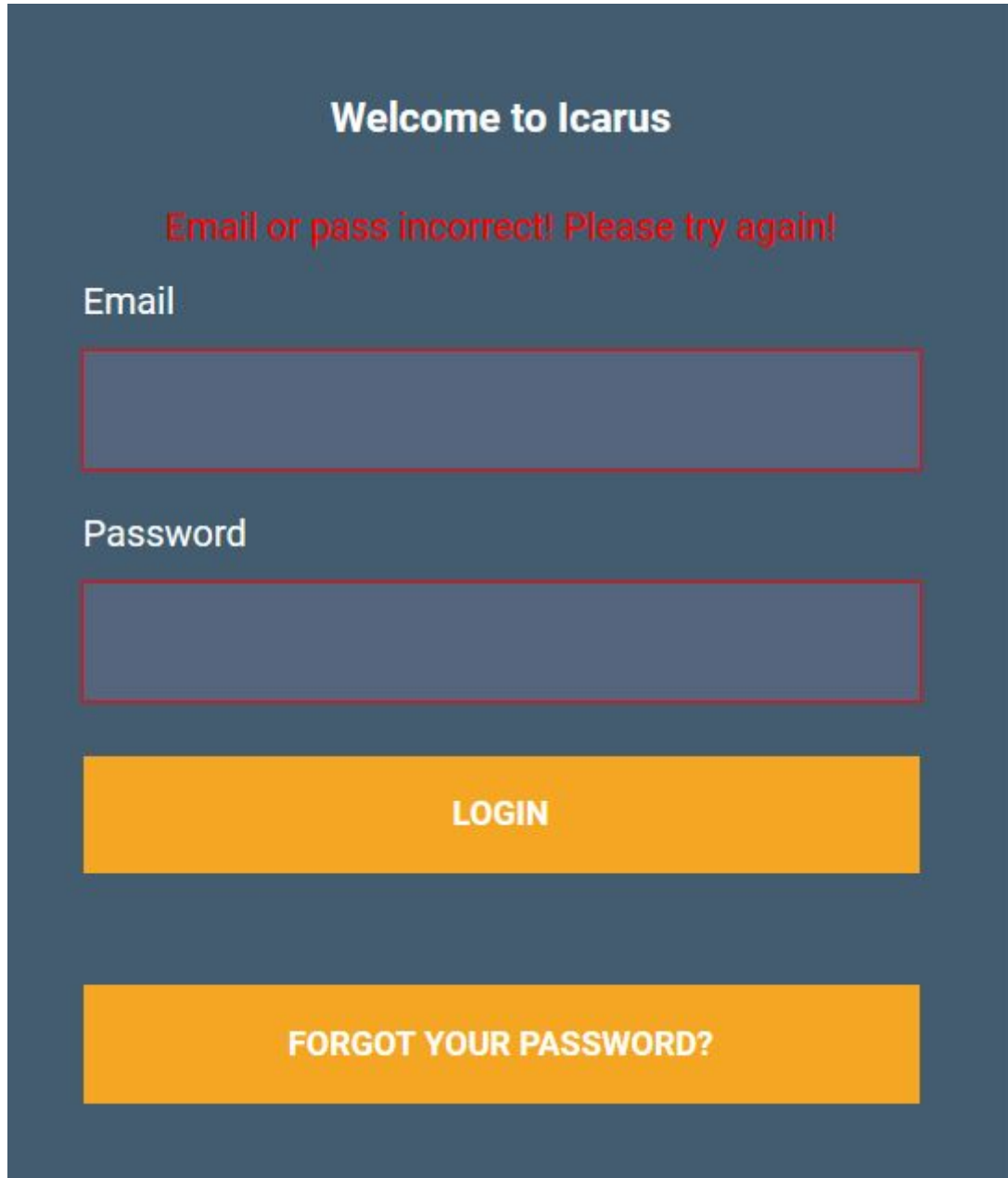
Password

.....

LOGIN

FORGOT YOUR PASSWORD?

Login Page Validation



Welcome to Icarus

Email or pass incorrect! Please try again!

Email

Password

LOGIN

FORGOT YOUR PASSWORD?

Welcome to Icarus

Email

Password



Please fill in this field.

LOGIN

FORGOT YOUR PASSWORD?

Dashboard Page



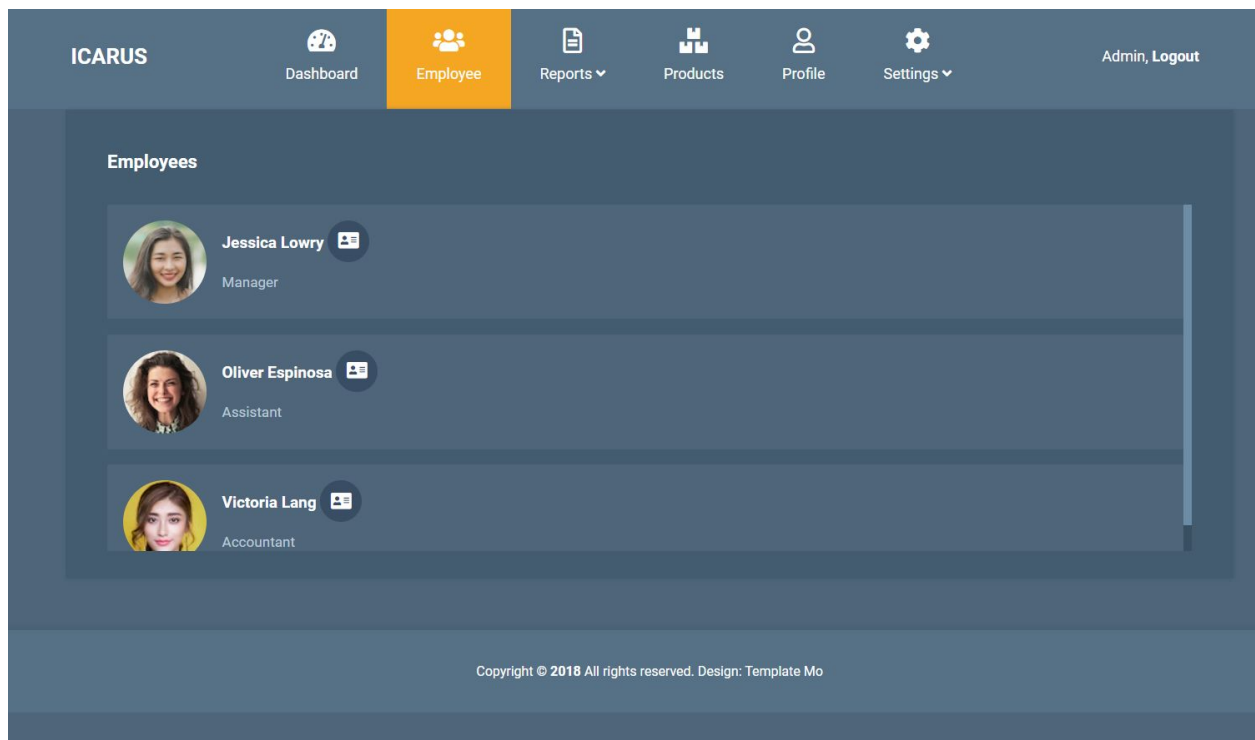
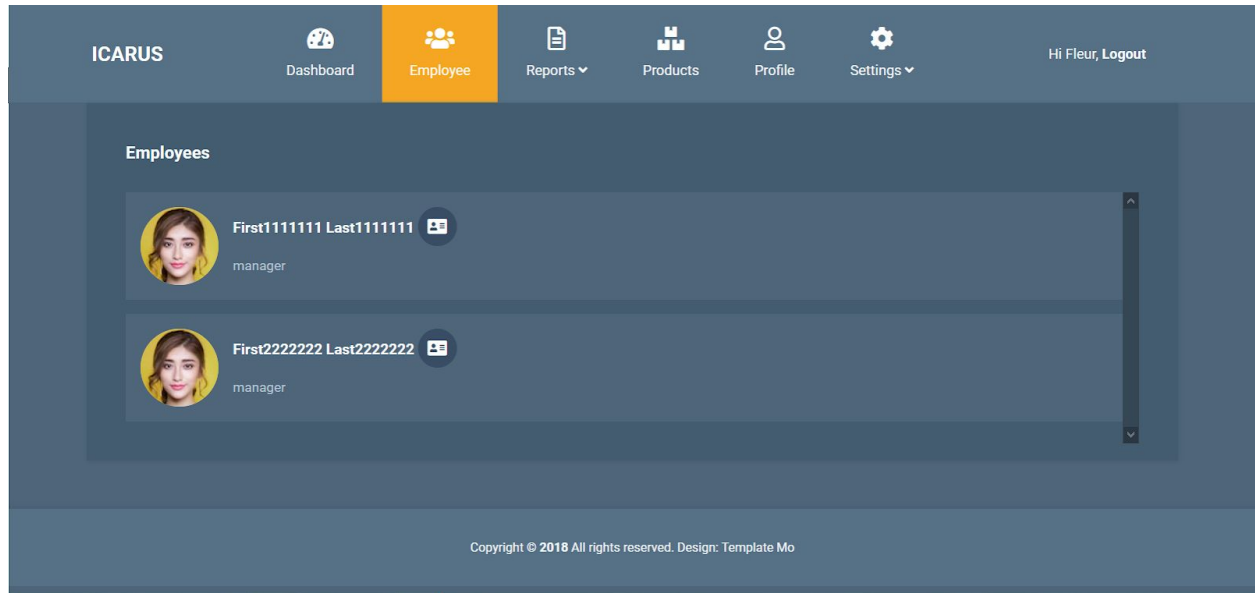
Sophie and 6 others sent you **product updates.**

6h ago.

Orders List

ORDER NO.	STATUS	OPERATORS	LOCATION	DISTANCE	START DATE	EST DELIVERY DUE
#122349	● Moving	Oliver Trag	London, UK	485 km	16:00, 12 NOV 2018	08:00, 18 NOV 2018
#122348	● Pending	Jacob Miller	London, UK	360 km	11:00, 10 NOV 2018	04:00, 14 NOV 2018
#122347	● Cancelled	George Wilson	London, UK	340 km	12:00, 22 NOV 2018	06:00, 28 NOV 2018
#122346	● Moving	William Aung	London, UK	218 km	15:00, 10 NOV 2018	09:00, 14 NOV 2018
#122345	● Pending	Harry Ryan	London, UK	280 km	15:00, 11 NOV 2018	09:00, 17 NOV 2018
#122344	● Pending	Michael Jones	London, UK	218 km	18:00, 12 OCT 2018	06:00, 18 OCT 2018

Employees Page



Order Report Page

ICARUS

Dashboard

Employee

Reports ▾

Products

Profile

Settings ▾

Hi Fleur, Logout

Orders List

Order NO.	Customer Id	Order Status	Payment Type	Quantity	Submission Date	Delivery Status	Estimated Delivery Due
#11111	1111	<div><div></div>PaidNDeli</div>	online	13	2020-11-25	InProgress	2020-01-21
#22222	2222	<div><div></div>PaidSelfDeli</div>	cash	13	2020-11-25	Delivered	2020-02-22
#33333	2222	<div><div></div>PaidDeliD</div>	cheque	13	2020-11-25	NotProc	2020-03-23

Filters

Order Status

Select Order Status ▾

Delivery Status

Select Delivery Status ▾

FILTER!

ICARUS

Dashboard

Employee

Reports ▾

Products

Profile

Settings ▾

Hi Fleur, Logout

Orders List

Order NO.	Customer Id	Order Status	Payment Type	Quantity	Submission Date	Delivery Status	Estimated Delivery Due
#22222	2222	<div><div></div>PaidSelfDeli</div>	cash	13	2020-11-25	Delivered	2020-02-22

Filters

Order Status

Select Order Status ▾

Select Order Status

Unpaid

Paid not delivered yet!


Paid and self delivered.


Paid and delivered completely.


Select Delivery Status ▾


FILTER!


ICARUS


Dashboard

Employee

Reports ▾

Products

Profile

Settings ▾

Hi Fleur, Logout

Orders List

Order NO.	Customer Id	Order Status	Payment Type	Quantity	Submission Date	Delivery Status	Estimated Delivery Due
#11111	1111	<div><div></div>PaidNDeli</div>	online	13	2020-11-25	InProgress	2020-01-21

Filters

Order Status

Select Order Status ▾

Select Delivery Status

Not Processed yet!

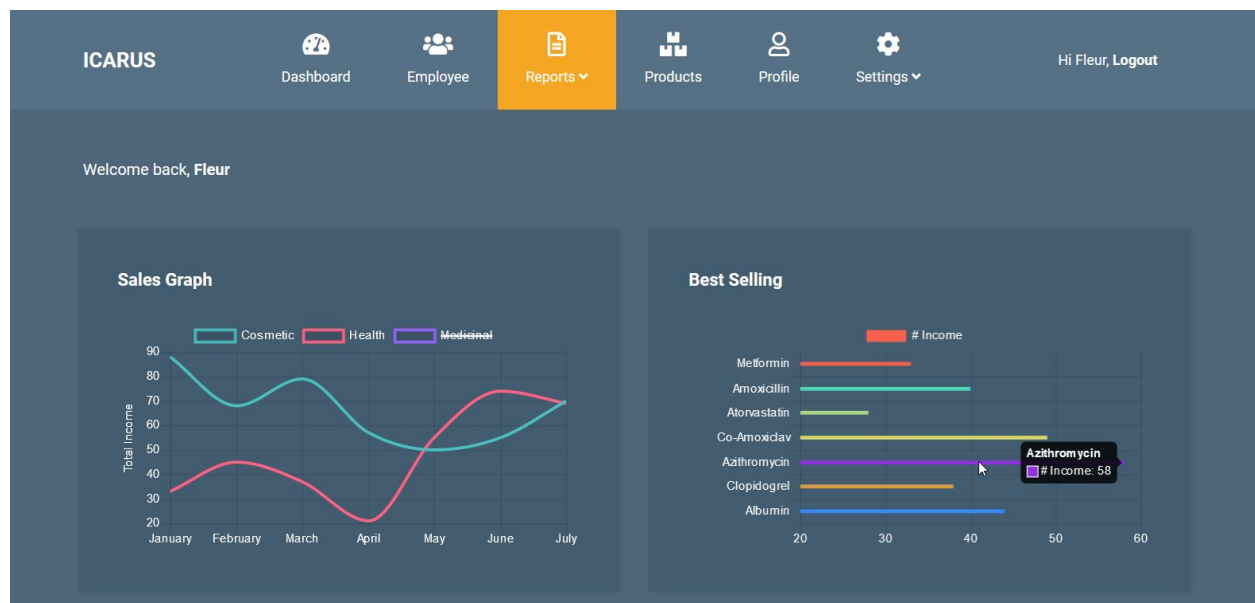
In Progress.

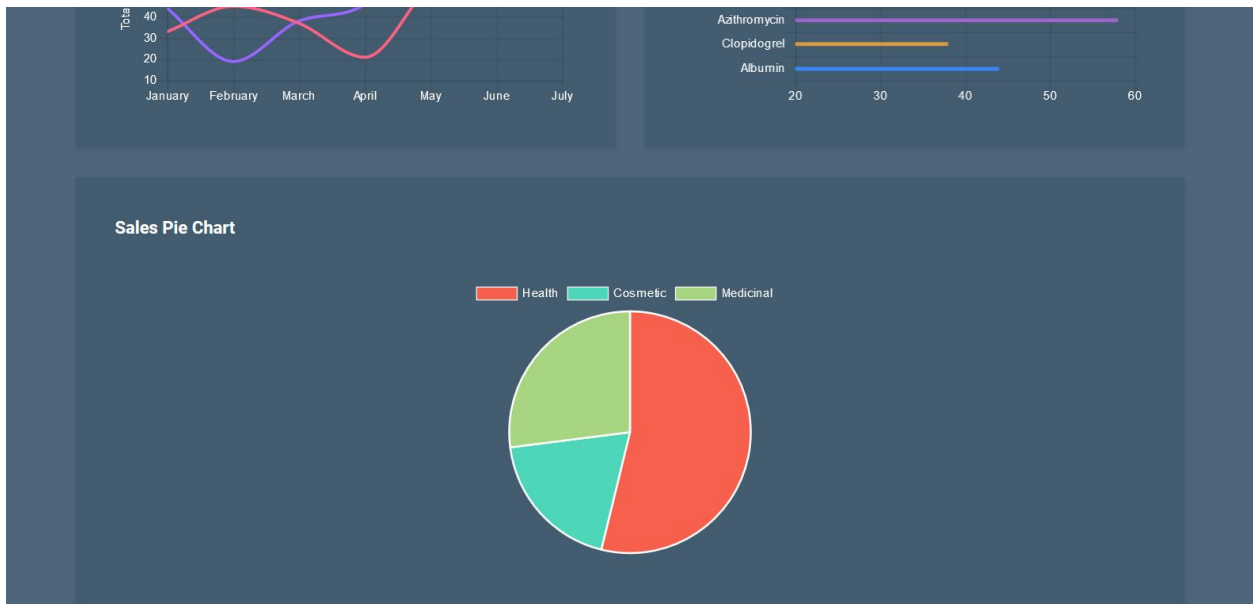
Delivered.

In Progress. ▾

FILTER!

Sales Report Page





Products Page

ICARUS

Dashboard

Employee

Reports ▾

Products

Profile

Settings ▾

Hi Fleur, Logout

	Name	Unit Price	In Stock	Type	Expiry Date	
<input checked="" type="checkbox"/>	goods111	111000	1000	Medi		
<input checked="" type="checkbox"/>	goods333	333000	2000	Cosm		
<input checked="" type="checkbox"/>	goods444	444000	3000	Heal		
<input checked="" type="checkbox"/>	goods-ui-ghazale	8745	585	Medi		

DELETE SELECTED PRODUCTS

Product Categories

Select Category ▾

FILTER!

ICARUS

Dashboard

Employee

Reports ▾

Products

Profile

Settings ▾

Hi Fleur, Logout

	Name	Unit Price	In Stock	Type	Expiry Date	
<input checked="" type="checkbox"/>	goods111	111000	1000	Medi		
<input checked="" type="checkbox"/>	goods-ui-ghazale	8745	585	Medi		

DELETE SELECTED PRODUCTS

Product Categories

Medicinal ▾

Select Category

Health

Cosmetic

Medicinal

Add Product Page

new product name

Product Id

January 2021

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

| I

ADD PRODUCT NOW!!

UPLOAD PRODUCT IMAGE

Category

Select Category

Product Id

6791

Origin Country

France

Unit Price

12000

Units In Stock

500

Produce Date

01/25/2020

Expire Date

01/25/2025

ADD PRODUCT NOW!!

UPLOAD PRODUCT IMAGE

Category

Select Category

Select Category


Health


Cosmetic


Medicinal


Account Page


ICARUS


Dashboard

Employee

Reports ▾


Products

Profile

Settings ▾

Hi Fleur, [Logout](#)

Account Picture



UPLOAD NEW PHOTO

Account Settings

Account Name

Fleur Masson

Account Email

fleur_masson@outlook.com

Password

••••••

Re-enter Password

Phone

+41-775-5525-07

توضیح و نصب ملزومات و پیش‌نیازها

Frontend

HTML

CSS

JQUERY

Backend

Flask

Flask یک میکرو فریم ورک برای توسعه وب است. این فریم ورک دارای هسته ای کوچک و ساده ولی با قابلیت انعطاف پذیری و گسترش با Plugin هایی از جمله Babel, CouchDB, MongoDB و ... می باشد.

لازم به ذکر است Micro به معنای این که تمامی برنامه های وب شما می بایست در یک فایل واحد پایتون قرار بگیرند نیست (اگرچه قاعدتا می تواند بدین سان هم باشد). همانطور که در بالاتر ذکر شد در این Framework، هسته ی اصلی به صورت کوچک و ساده طراحی شده ولی می توان به راحتی آن را گسترش داد.

فریم ورک Flask از آخرین نسخه از پایتون 3، 2.7 و PyPy پشتیبانی می کند، در هنگام نصب این فریم ورک پیش نیازهای ذیل به صورت اتوماتیک نصب می گردند:

- Werkzeug
- Jinja
- MarkupSafe
- ItsDangerous
- Click

Virtual Environment

از یک Virtual Environment برای مدیریت وابستگی های یک پروژه در محیط تولید و عملیات استفاده می شود.

شاید این سوال پیش بیاید که چرا می بایست از یک Virtual Environment استفاده نمود؟

هرچقدر تعداد پروژه های پایتون بیشتری داشته باشیم، به احتمال بسیار زیاد می بایست از نسخه های مختلف Library های پایتون استفاده نمود و نکته ی مهم اینجاست که نسخه های مختلفی از یک Library می تواند باعث عدم اجرای صحیح پروژه های دیگر شود در همین راستا Virtual Environment ها محیط های مستقلی از Library های پایتون هستند که فایل های نصب شده برای هر پروژه را با پروژه ی دیگری متفاوت خواهند نمود.

ایجاد Virtual Environment

- لینوکس، نسخه 3 پایتون:

```
mkdir myproject
```

```
cd myproject
```

```
python3 -m venv venv
```

● ویندوز، نسخه 3 پایتون:

```
py -3 -m venv venv
```

فعال سازی Virtual Environment

پیش از شروع کار بر روی پروژه، می بایست Virtual Environment ایجاد شده را با استفاده از دستور ذیل فعال نمود:

● لینوکس:

```
venv/bin/activate
```

● ویندوز:

```
venv\Scripts\activate
```

نصب Flask

پس از فعال سازی Virtual Environment، می بایست Flask را با استفاده از دستور ذیل نصب نمایید:

```
pip install Flask
```

و در اغلب موارد در زمان کار با پایگاه داده نیز استفاده می‌شود. `methods = ['POST']` برای ارسال داده‌ها به سرور استفاده می‌شود. ضمناً باید مطمئن شویم که محیط مجازی نیز برای این اپلیکیشن راه‌اندازی شده است.

ایجاد یک پایگاه داده PostgreSQL

PgAdmin نرم افزار تحت وب به منظور مدیریت پایگاه داده های PostgreSQL می باشد. PgAdmin به زبان Python و jQuery همراه با Bootstrap نوشته شده است که می توان آن را بر روی پلتفرم های مختلف مانند لینوکس، مک او اس و ویندوز نصب و راه اندازی کرد.

ما قصد داریم برای ایجاد یک مخزن داده از یک سیستم مدیریت پایگاه استفاده کنیم. از این رو از PostgreSQL به این منظور استفاده می‌کنیم. اگر روی سیستم ویندوز یا Mac کار می‌کنید، فایل‌های نصبی برای این پایگاه داده در وبسایت رسمی آن وجود دارد. در صورتی که از سیستم لینوکس استفاده می‌کنید، می‌توانید با مراجعه به Ubuntu Software نسخه 16.04 یا جدیدتر به دنبال اپلیکیشن pgAdmin III بگردید. با این وجود اگر از اوبونتو 15.10 یا قدیمی‌تر استفاده می‌کنید.

ایجاد کاربر

زمانی که pgAdmin 3 را نصب کردید باید یک کاربر ایجاد کرده و رمز عبوری برای آن تعیین کنید.