E-commerce Website

By

Najia Anjum 2018-1-60-125
Md. Abu Horayra 2017-2-60-077
Md. BakiBillah Chowdhuryy 2017-2-60-146

An E-commerce website for selling Electronics products.

Problems

Solution

Developing an website with Python(Django), HTML, CSS and PostgreSQL so that customers can browse products and do shopping online. It will also have a delivery management system.

Selling Electronic products Online

Need a website for selling products
Here are the modules .

- Authentication and Authorization
- Products Catalog
- Cart and Checkout
- Order Management
- Delivery Management

Requirements

Functional Requirements

- Registration System (customer, delivery man)
- Login
- Forget password
- Reset password
- Add or delete order
- Product Visibility

Non Functional Requirements

- Server/site should be available all time
- Adaptability
- The software must be adaptive
- The software must be flexible
- User friendly
- Security
- Server must be able to handle a lot of request at a time
- The website should be compatible with all the browsers.
- All requests must respond in less than 3 sec.

Feasibility Study

Technical Feasibility:

- With the latest high end tech tool like Django, we can create an extremely powerful website within a short deadline.

- Combining latest tools like HTML, CSS and PostgreSQL and standard computer our website will stands the test of time in terms of performance and great design.

Economic Feasibility:

- Due to pandemic, people don't need to get out of their house to buy products. People can buy products through our while staying home. So, owning a site like this can be a strong source of passive income.

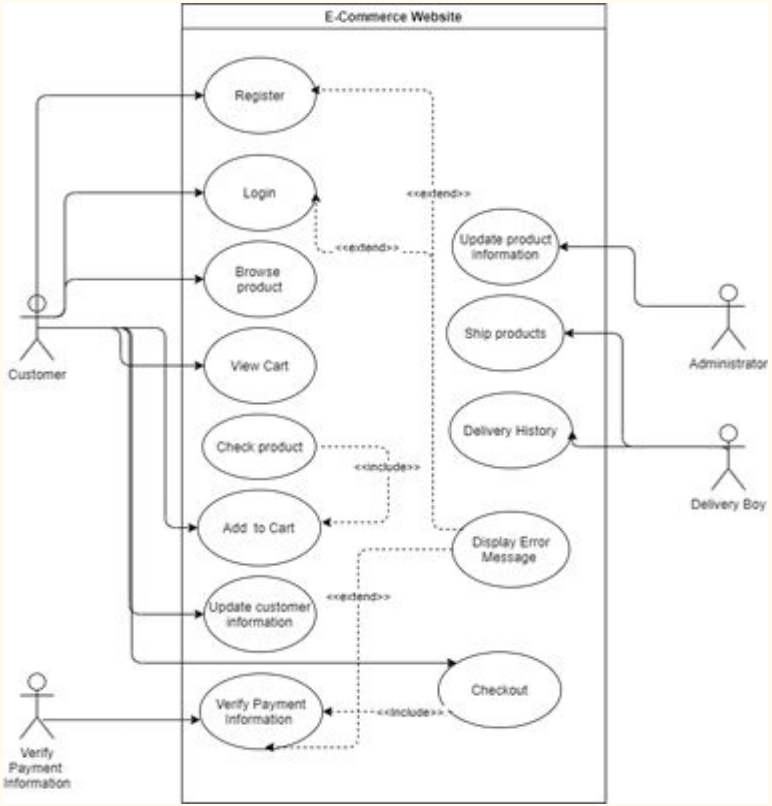
- Moreover, we are helping the customers get their valuable necessities without having to risk their lives and getting out on the streets.

- Our cost is maintain database which is very low and maintain workers.

Operational Feasibility:

• As we have a wider variety of products due to owing a virtual store, we will be able to provide service and products to customers with a higher discount than a retail shop.

Use Case Diagram



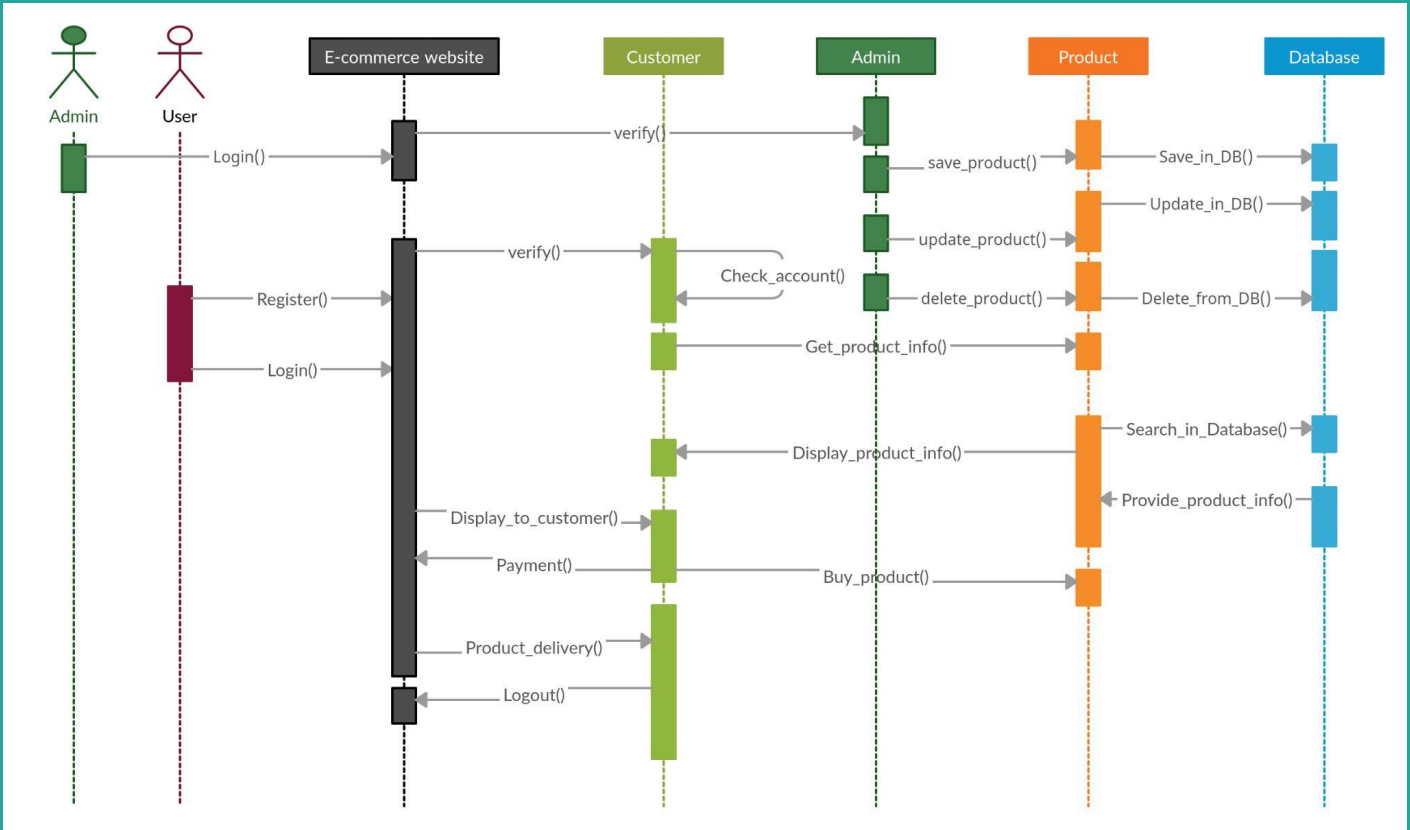
Browse Products:

The customer can request to view the product in the product catalog. The system will display the information about products of selected categories.

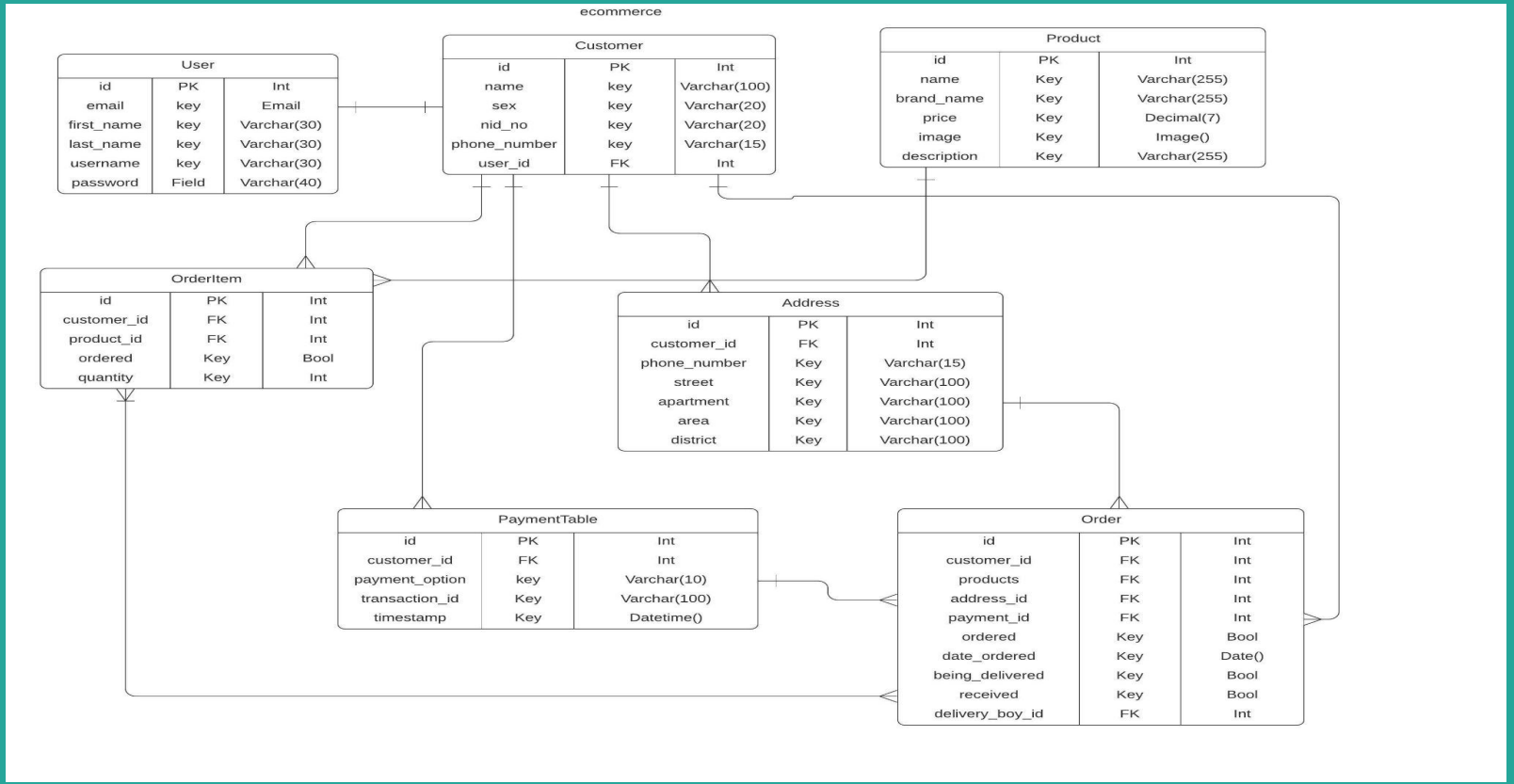
Add to Cart:

Customers can add one of the multiple products in his/her cart. The system stores and tracks the information about the products.

Sequence Diagram



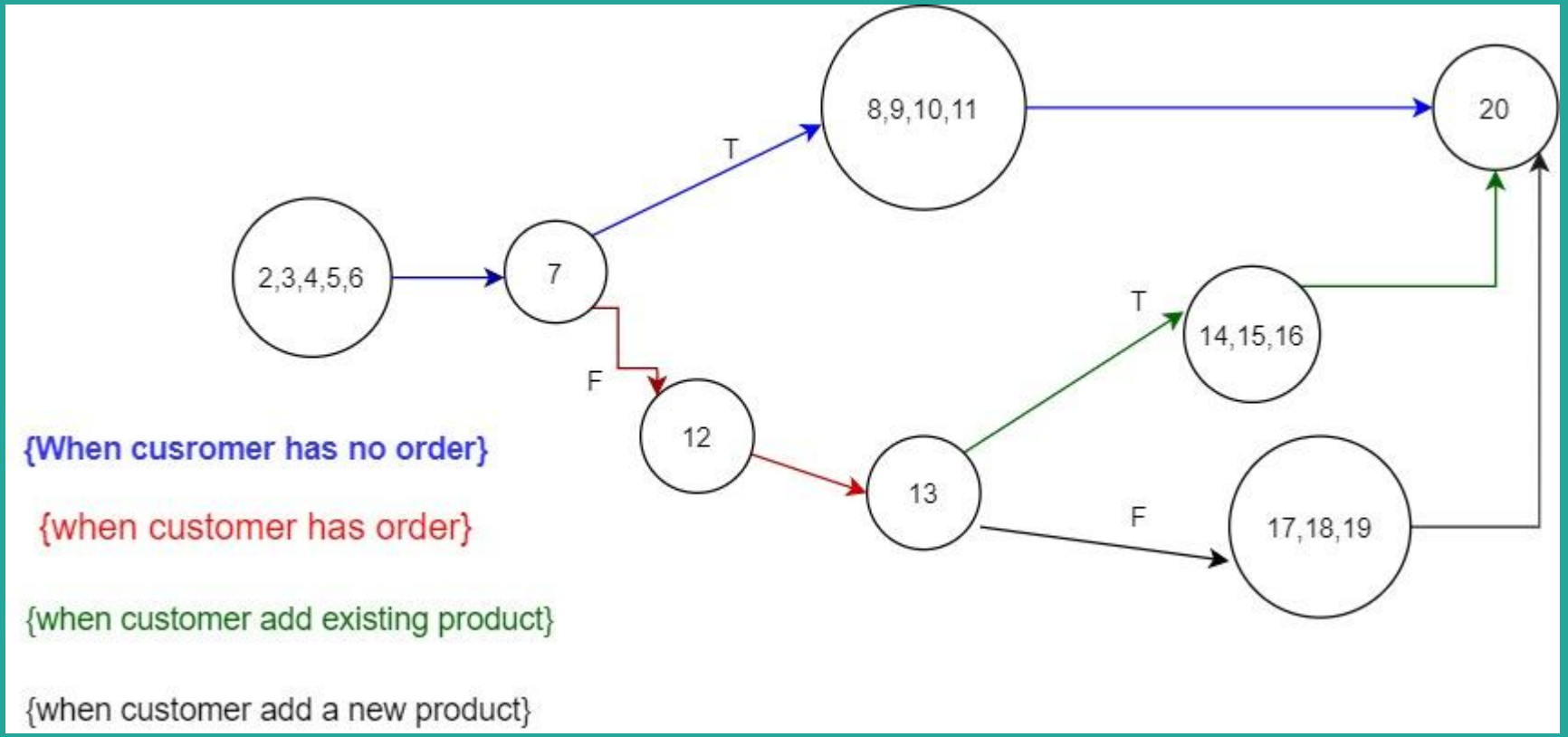
ER Diagram


ecommerce
User
id PK Int
email key Email
first_name key Varchar(30)
last_name key Varchar(30)
username key Varchar(30)
password Field Varchar(40)
Customer
id PK Int
name key Varchar(100)
sex key Varchar(20)
nid_no key Varchar(20)
phone_number key Varchar(15)
user_id FK Int
Product
id PK Int
name Key Varchar(255)
brand_name Key Varchar(255)
price Key Decimal(7)
image Key Image()
description Key Varchar(255)
OrderItem
id PK Int
customer_id FK Int
product_id FK Int
ordered Key Bool
quantity Key Int
Address
id PK Int
customer_id FK Int
phone_number Key Varchar(15)
street Key Varchar(100)
apartment Key Varchar(100)
area Key Varchar(100)
district Key Varchar(100)
PaymentTable
id PK Int
customer_id FK Int
payment_option key Varchar(10)
transaction_id Key Varchar(100)
timestamp Key Datetime()
Order
id PK Int
customer_id FK Int
products FK Int
address_id FK Int
payment_id FK Int
ordered Key Bool
date_ordered Key Date()
being_delivered Key Bool
received Key Bool
delivery_boy_id FK Int

Test Code

```python
def add_to_cart(request, id):
    current_user = request.user
    product = get_object_or_404(Product, id=id)
    customer = Customer.objects.get(id=current_user.id)
    order_item, created = OrderItem.objects.get_or_create(product=product, customer=customer, ordered=False)
    order_qs = Order.objects.filter(**{'customer': customer, 'ordered': False}).first()
    if order_qs is None:
        order = Order.objects.create(**{'customer': customer, 'date_ordered': str(date.today())})
        order.products.add(order_item)
        order.save()
        messages.info(request, "This item was added to your cart.")
    else:
        if order_qs.products.filter(product__id=product.id).exists():
            order_item.quantity += 1
            order_item.save()
            messages.info(request, "This item quantity was updated")
        else:
            order_qs.products.add(order_item)
            messages.info(request, "This item was added to your cart.")
    return redirect("order_summary")
```

White Box Testing



Thank You