UIT2602 WEB PROGRAMMING
Ex. No.: 1 - Web Stock Maintenance System

Name: S. Selcia
Reg. No.: 3122215002098
Section: IT - B

1. Aim:

The aim of this web application is to create a Stock Maintenance System that facilitates the management of product details, purchases, sales, and stock levels. The primary objectives include providing a user-friendly interface for data entry, real-time updates on stock movements using HTML and DHTML, incorporating various web technologies

2. Required Web Tools and Methodology:

To achieve the goals of the Stock Maintenance System, the following tools and methodologies were employed:

- Web Development Framework: Flask (Python)
- Database: MongoDB
- Frontend: HTML, CSS, Jinja2 (template engine for Flask)
- Charting Library: Chart.js
- Version Control: Git

The development process includes Flask handling the backend logic, MongoDB storing the data, and HTML/CSS rendering the frontend.

3. Implementation Procedure:

a. Database Design:
    - Defined MongoDB collections for products, purchases, sales, and stock.
    - Established relationships between these collections.

b. Backend Development:
    - Use Flask to create routes for adding and retrieving data.
    - Implement functions to update stock levels based on purchases and sales.

c. Frontend Development:
    - Create HTML templates for product, purchase, sales, and stock pages.
    - Use CSS for styling and layout.
    - Integrated Chart.js for dynamic statistics.

d. User Interface:
   - Implemented internal hyperlinking for seamless navigation between different sections of the web application.
   - Designed forms for inputting product details, purchase information, and sales data.
   - Implemented tables to display structured data, such as product details and stock information.
   - Incorporate dynamic charts for statistics.

4. Code Snippets:

- `app.py`

```
from flask import Flask, render_template, request, redirect, url_for
from flask_pymongo import PyMongo
from datetime import datetime

app = Flask(__name__)
app.config['MONGO_URI'] =
'mongodb+srv://selcia2110605:uUXoiQVyFTSZI0qc@stockmaintenance.ztbi9gj.mongodb.net/users_database'
mongo = PyMongo(app)

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

@app.route('/product')
def product():
    products = mongo.db.products.find()
    return render_template('product.html', products=products)

@app.route('/add_product', methods=['POST'])
def add_product():
    if request.method == 'POST':
        code = request.form['code']
        name = request.form['name']
        opening_stock = int(request.form['opening_stock'])
        prices = request.form['prices']

        # Insert the product into MongoDB
        mongo.db.products.insert_one({
            'code': code,
            'name': name,
            'opening_stock': opening_stock,
            'prices': prices
```

```python
        })
        products.append({"code": code, "name": name, "opening_stock": opening_stock, "prices": 
prices})

        # Initialize stock details for the new product
        initialize_stock(code, opening_stock)
        return redirect(url_for('product'))

@app.route('/purchase')
def purchase():
    purchases = mongo.db.purchases.find()
    return render_template('purchase.html', purchases=purchases)

@app.route('/add_purchase', methods=['POST'])
def add_purchase():
    if request.method == 'POST':
        product_code = request.form['product_code']
        quantity = int(request.form['quantity'])
        price = request.form['price']
        date = datetime.now()
        # Insert the purchase into MongoDB
        mongo.db.purchases.insert_one({
            'product_code': product_code,
            'quantity': quantity,
            'price': price,
            'date': date
        })

        update_stock(product_code, quantity, 0)
        return redirect(url_for('purchase'))

def update_stock(product_code, purchase_quantity, sales_quantity):
    # Get the current stock details for the product
    stock_details = mongo.db.stocks.find_one({'product_code': product_code})

    # Update the stock details
    new_purchase_stock = stock_details.get('purchase_stock', 0) + purchase_quantity
    new_sales_stock = stock_details.get('sales_stock', 0) + sales_quantity
    new_current_stock = stock_details.get('opening_stock', 0) + new_purchase_stock - new_sales_stock

    # Update the stock details in the stocks collection
    mongo.db.stocks.update_one(
        {'product_code': product_code},
        {'$set': {
            'purchase_stock': new_purchase_stock,
            'sales_stock': new_sales_stock,
            'current_stock': new_current_stock,
        }}
    )

@app.route('/sales')
def sales():
    sales = mongo.db.sales.find()
    return render_template('sales.html', sales=sales)

@app.route('/add_sales', methods=['POST'])
def add_sales():
    if request.method == 'POST':
        date = datetime.now()
        customer_name = request.form['customer_name']
        product_code = request.form['product_code']
        quantity = int(request.form['quantity'])
        price = request.form['price']

        mongo.db.sales.insert_one({
            'customer_name': customer_name,
            'product_code': product_code,
            'quantity': quantity,
            'price': price,
            'date': date
        })

        update_stock(product_code, 0, quantity)

        return redirect(url_for('sales'))

def initialize_stock(product_code, opening_stock):
    # Check if stock details already exist for the product
    existing_stock = mongo.db.stocks.find_one({'product_code': product_code})

    if not existing_stock:
        # If no stock details exist, insert a new document with the initial stock values
        mongo.db.stocks.insert_one({
            'product_code': product_code,
            'opening_stock': opening_stock,
            'purchase_stock': 0,
            'sales_stock': 0,
            'current_stock': opening_stock,
        })

@app.route('/stock')
def stock():
    stock_details = mongo.db.stocks.find()
    return render_template('stock.html', stock_details=stock_details)

if __name__ == '__main__':
    app.run(debug=True)
```

- **index.html**

```
<!DOCTYPE html>
<html>
<head>
    <title>Web Stock Maintenance</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <!-- Include Chart.js from CDN -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <nav>
        <a href="{{ url_for('index') }}" >Home</a>
        <a href="{{ url_for('product') }}" >Products</a>
        <a href="{{ url_for('purchase') }}">Purchase</a>
        <a href="{{ url_for('sales') }}">Sales</a>
        <a href="{{ url_for('stock') }}">Stock</a>
    </nav>
    <main>
        <section id="dashboard">
            <h2>Welcome to the Stock Maintenance System</h2>
            <div>
                <h3>Recent Activity</h3>
                <ul>
                    <li>Purchase of Product A - 50 units</li>
                    <li>Sale of Product B - 20 units</li>
                </ul>
            </div>
            <div>
                <h3>Statistics</h3>
                <canvas id="statisticsChart" width="400" height="100"></canvas>
                <p>Total Products: 100</p>
                <p>Total Purchases: 500</p>
                <p>Total Sales: 300</p>
            </div>
        </section>
    </main>
    <script>
    // Chart.js initialization
    var ctx = document.getElementById('statisticsChart').getContext('2d');
    var myChart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: ['Products', 'Purchases', 'Sales'],
            datasets: [{
                label: 'Statistics',
                data: [100, 500, 300],
                backgroundColor: [
                    'rgba(75, 192, 192, 0.2)',
                    'rgba(255, 99, 132, 0.2)',
                    'rgba(54, 162, 235, 0.2)'
                ],
                borderColor: [
                    'rgba(75, 192, 192, 1)',
                    'rgba(255, 99, 132, 1)',
                    'rgba(54, 162, 235, 1)'
                ],
                borderWidth: 1
            }]
        },
        options: {
            scales: {
                y: {
                    beginAtZero: true
                }
            }
        }
    });
</script>
</body>
</html>
</body>
</html>
```

- **home.html**

```
<!DOCTYPE html>
<html>
<head>
    <title>Web Stock Maintenance</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
    <nav>
        <a href="{{ url_for('index') }}" >Home</a>
        <a href="{{ url_for('product') }}" >Products</a>
        <a href="{{ url_for('purchase') }}">Purchase</a>
        <a href="{{ url_for('sales') }}">Sales</a>
        <a href="{{ url_for('stock') }}">Stock</a>
    </nav>
    <main>
        {% block content %}
        {% endblock %}
    </main>
</body>
</html>
```

- **product.html**

```
{% extends 'home.html' %}

{% block content %}
    <section id="product">
        <h2>Product Details</h2>

        <table>
            <thead>
                <tr>
                    <th>Product Code</th>
                    <th>Product Name</th>
                    <th>Opening Stock</th>
                    <th>Prices</th>
                </tr>
            </thead>
            <tbody>
                {% for product in products %}
                    <tr>
                        <td>{{ product.code }}</td>
                        <td>{{ product.name }}</td>
                        <td>{{ product.opening_stock }}</td>
                        <td>{{ product.prices }}</td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>

        <form action="{{ url_for('add_product') }}" method="post">
            <label for="code">Product Code:</label>
            <input type="text" id="code" name="code" required>
            <label for="name">Product Name:</label>
            <input type="text" id="name" name="name" required>
            <label for="opening_stock">Opening Stock:</label>
            <input type="number" id="opening_stock" name="opening_stock" required>
            <label for="prices">Prices:</label>
            <input type="text" id="prices" name="prices" required>
            <button type="submit">Add Product</button>
        </form>
    </section>
{% endblock %}
```

- **purchase.html**

```
{% extends 'home.html' %}

{% block content %}
    <section id="purchase">
```

```
        <h2>Purchase Details</h2>
        <table>
            <thead>
                <tr>
                    <th>Date</th>
                    <th>Product Code</th>
                    <th>Quantity</th>
                    <th>Price</th>
                </tr>
            </thead>
            <tbody>
                {% for purchase in purchases %}
                    <tr>
                        <td>{{ purchase.date }}</td>
                        <td>{{ purchase.product_code }}</td>
                        <td>{{ purchase.quantity }}</td>
                        <td>{{ purchase.price }}</td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>

        <form action="{{ url_for('add_purchase') }}" method="post">
            <label for="product_code">Product Code:</label>
            <input type="text" id="product_code" name="product_code" required>
            <label for="quantity">Quantity:</label>
            <input type="number" id="quantity" name="quantity" required>
            <label for="price">Price:</label>
            <input type="text" id="price" name="price" required>
            <button type="submit">Add Purchase</button>
        </form>
    </section>
{% endblock %}
```

- **sales.html**

```
{% extends 'home.html' %}

{% block content %}
    <section id="sales">
        <h2>Sales Details</h2>
        <form action="{{ url_for('add_sales') }}" method="post">
            <label for="date">Date:</label>
            <input type="date" id="date" name="date" required>
            <label for="customer_name">Customer Name:</label>
            <input type="text" id="customer_name" name="customer_name" required>
            <label for="product_code">Product Code:</label>
            <input type="text" id="product_code" name="product_code" required>
            <label for="quantity">Quantity:</label>
            <input type="number" id="quantity" name="quantity" required>
            <label for="price">Price:</label>
            <input type="text" id="price" name="price" required>
            <button type="submit">Add Sales</button>
        </form>
        <table>
            <thead>
                <tr>
                    <th>Date</th>
                    <th>Customer Name</th>
                    <th>Product Code</th>
                    <th>Quantity</th>
                    <th>Price</th>
                </tr>
            </thead>
            <tbody>
                {% for sale in sales %}
                    <tr>
                        <td>{{ sale.date }}</td>
                        <td>{{ sale.customer_name }}</td>
                        <td>{{ sale.product_code }}</td>
                        <td>{{ sale.quantity }}</td>
                        <td>{{ sale.price }}</td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>
    </section>
{% endblock %}
```

- **stock.html**

```
{% extends 'home.html' %}

{% block content %}
    <section id="stock">
        <h2>Stock Details</h2>
        <table>
            <thead>
                <tr>
                    <th>Product ID</th>
                    <th>Opening Stock</th>
                    <th>Purchase Stock</th>
                    <th>Sales Stock</th>
                    <th>Current Stock</th>
                </tr>
            </thead>
            <tbody>
                {% for stock_detail in stock_details %}
```

```
                    <tr>
                        <td>{{ stock_detail.product_code }}</td>
                        <td>{{ stock_detail.opening_stock }}</td>
                        <td>{{ stock_detail.purchase_stock }}</td>
                        <td>{{ stock_detail.sales_stock }}</td>
                        <td>{{ stock_detail.current_stock }}</td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>
    </section>
{% endblock %}
```

- **styles.css**

```
body,h1,h2,h3,p,ul,ol,li {
  margin: 0;
  padding: 0;
}
body {
  font-family: system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto,
    Oxygen, Ubuntu, Cantarell, "Open Sans", "Helvetica Neue", sans-serif;
  line-height: 1.6;
  background-color: #f5f5f5;
  color: #333;
}
nav {
  background: linear-gradient(to right, #333, #555);
  color: #fff;
  padding: 10px;
}
nav a {
  color: #fff;
  text-decoration: none;
  margin-right: 10px;
  text-align: center;
  padding: 14px 16px;
  border-radius: 4px;
  transition: background-color 0.3s, color 0.3s;
}
nav a:hover {
  background-color: #ddd;
  color: #333;
}
main {
  padding: 20px;
}
#product,#purchase,#sales,#stock {
  background-color: #fff;
```

```css
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);
  margin-top: 20px;
}
table {
  width: 100%;
  border-collapse: collapse;
  margin-top: 20px;
}
table,th,td {
  border: 2px solid #ddd;
}
th,td {
  padding: 12px;
  text-align: left;
}
th {
  background: linear-gradient(to right, #333, #555);
  color: white;
}
.logo-svg {
  width: 50px;
  height: 50px;
  margin-left: auto;
}
form {
  margin-top: 20px;
}
label {
  display: block;
  margin-bottom: 8px;
}
input,button {
  margin-bottom: 12px;
  padding: 10px;
  border: 1px solid #ddd;
  border-radius: 4px;
  transition: border-color 0.3s;
}
input:focus,button:focus {
  border-color: #555;
}
button {
  background: linear-gradient(to right, #333, #555);
  color: #fff;
  cursor: pointer;
}
```

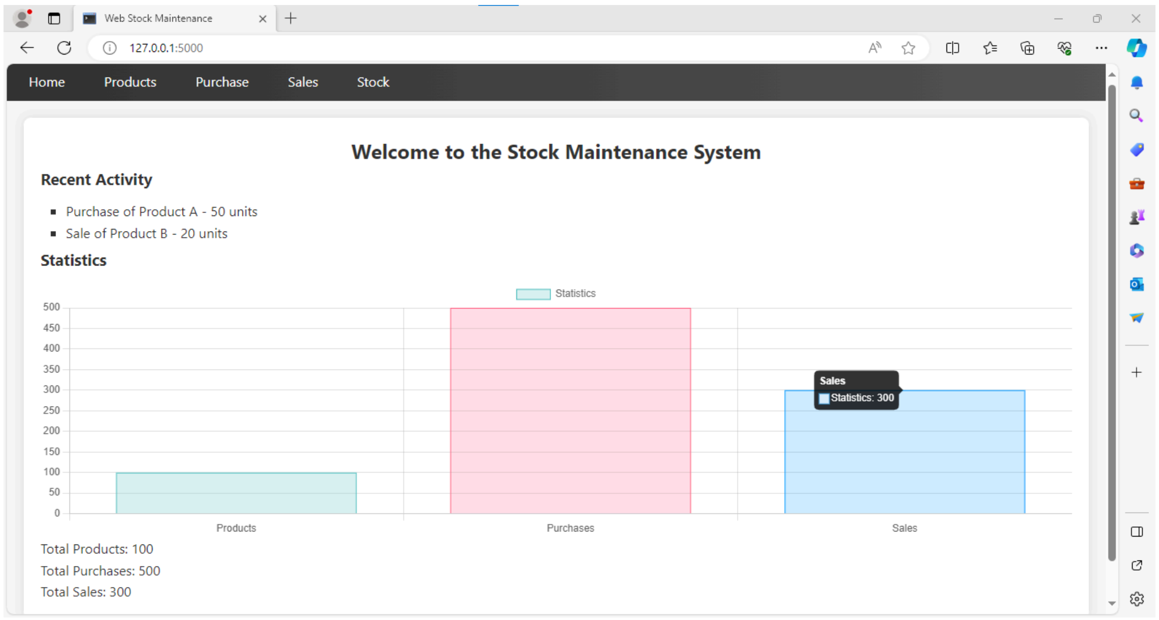
```css
button:hover {
  background: linear-gradient(to right, #555, #777);
}
#dashboard {
  background-color: #fff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  margin-bottom: 20px;
}
h2 {
  color: #333;
  text-align: center;
}
main {
  padding: 20px;
}
h3 {
  color: #333;
  margin-bottom: 10px;
}
canvas {
  margin-top: 0px;
}
ul {
  list-style-type: square;
  padding: 0;
  margin-bottom: 5px;
}
li {
  margin-left: 30px;
}
```

5. Output:

The web application provides a visually appealing and user-friendly interface for managing stock-related information. Users can add and view product details, record purchases and sales.

- index.html: Overview with recent activities and statistics
- product.html: Displays and manages product details.
- purchase.html: Handles purchase data input and display.
- sales.html: Manages sales data input and display.
- stock.html: Displays and updates stock details.
- home.html: Home page with navigation links.
- styles.css: Stylesheet defining the visual appearance.

Home Page



Welcome to the Stock Maintenance System

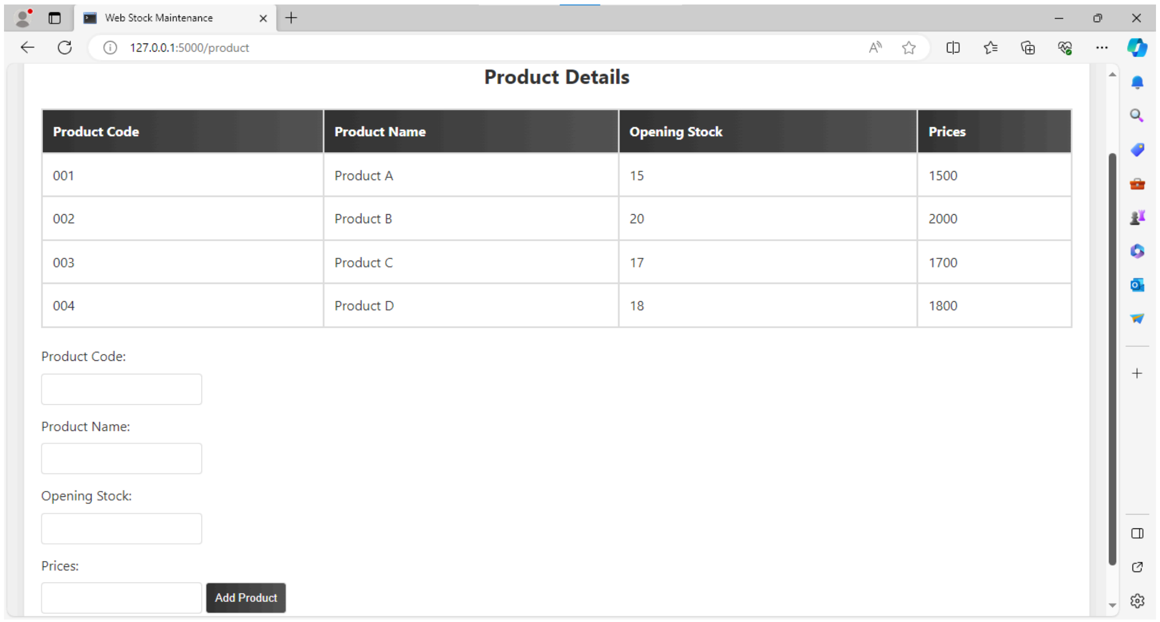
Recent Activity

- Purchase of Product A - 50 units
- Sale of Product B - 20 units

Statistics

Total Products: 100
Total Purchases: 500
Total Sales: 300

Products Page



Product Details

Product Code	Product Name	Opening Stock	Prices
001	Product A	15	1500
002	Product B	20	2000
003	Product C	17	1700
004	Product D	18	1800

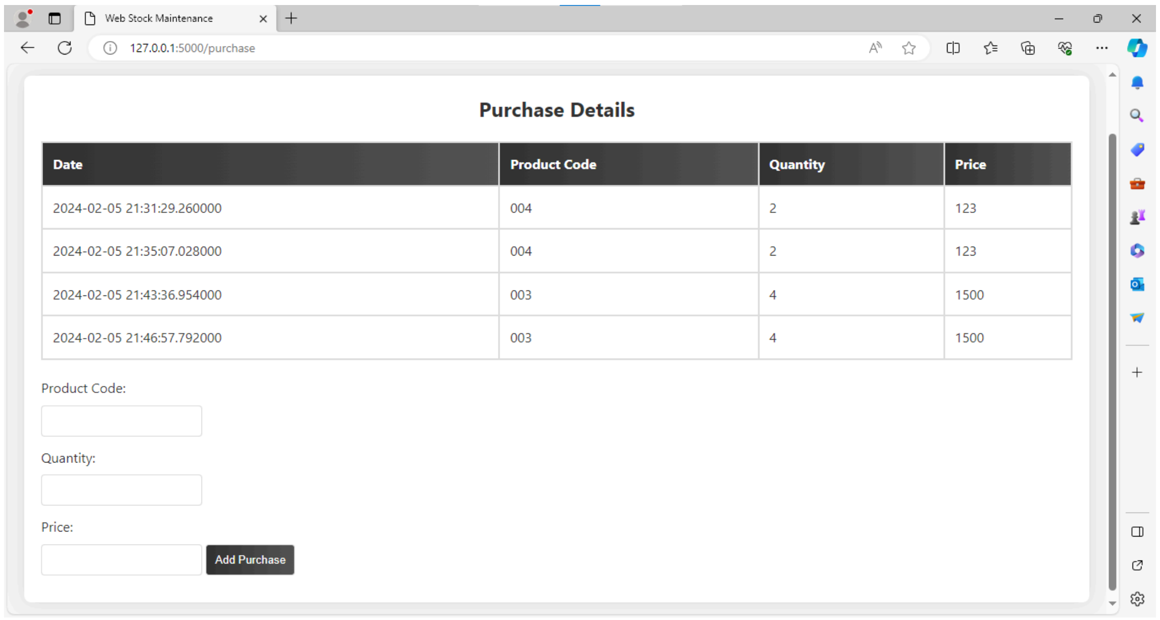
Product Code:

Product Name:

Opening Stock:

Prices:

Purchases Page



Purchase Details

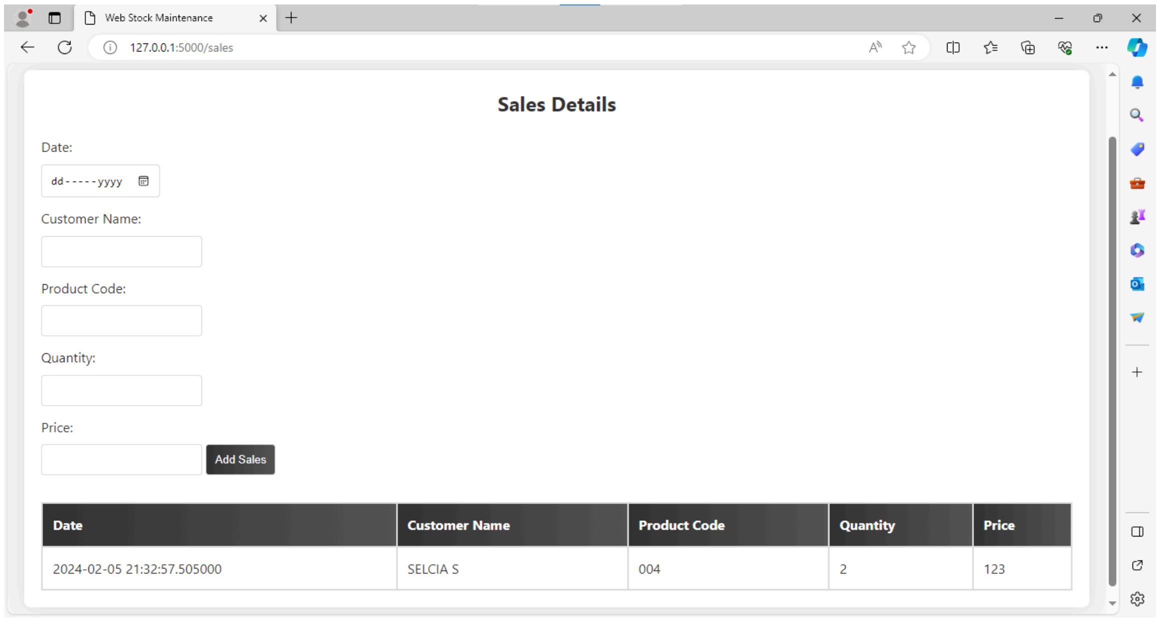
Date	Product Code	Quantity	Price
2024-02-05 21:31:29.260000	004	2	123
2024-02-05 21:35:07.028000	004	2	123
2024-02-05 21:43:36.954000	003	4	1500
2024-02-05 21:46:57.792000	003	4	1500

Product Code:

Quantity:

Price:

Sales Page



Sales Details

Date:

dd-----yyyy

Customer Name:

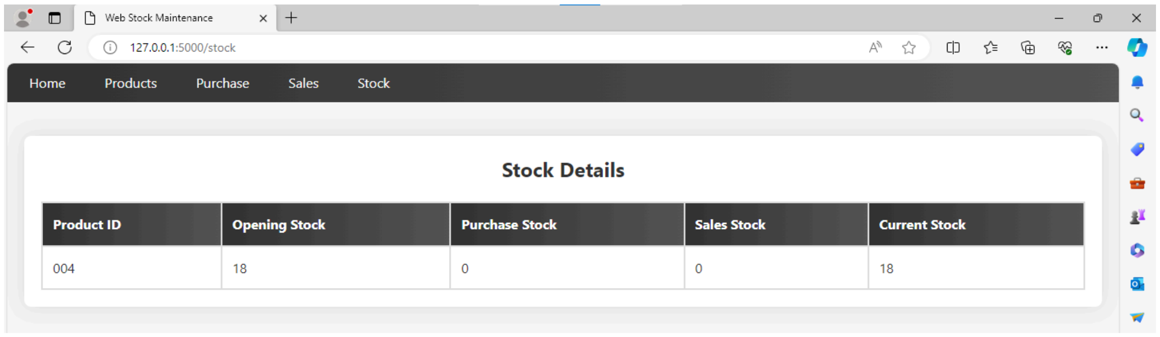
Product Code:

Quantity:

Price:

Date	Customer Name	Product Code	Quantity	Price
2024-02-05 21:32:57.505000	SELCIA S	004	2	123

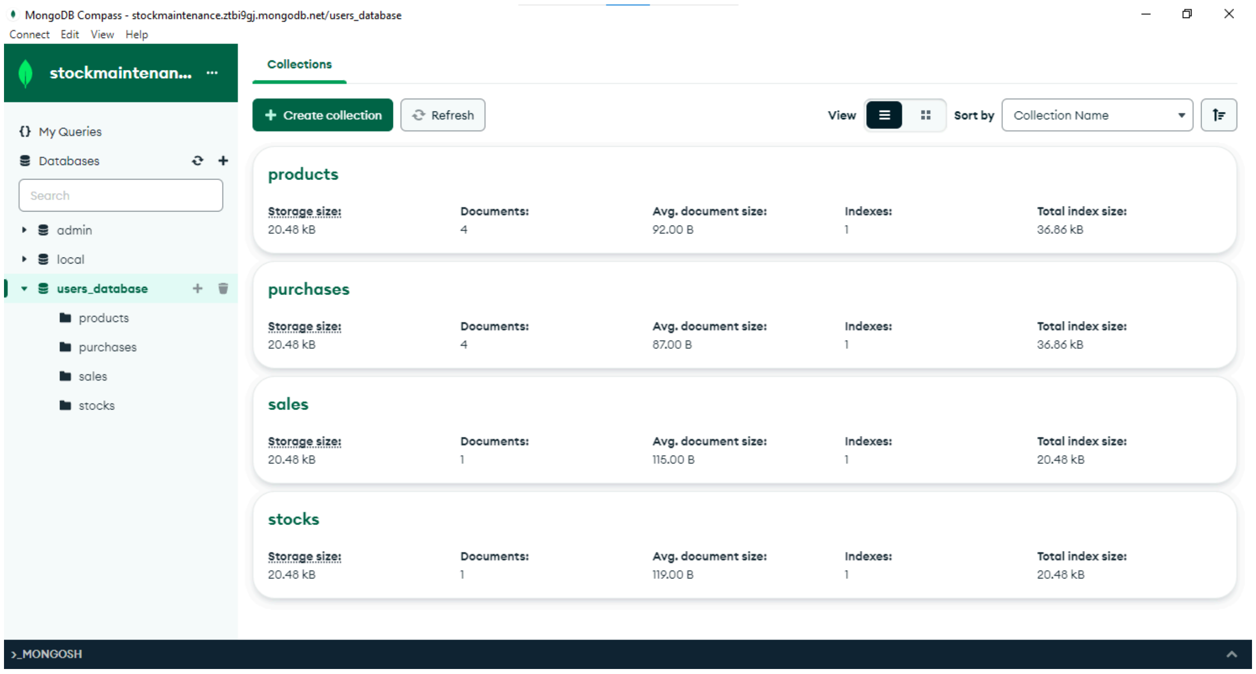
Stock Page:



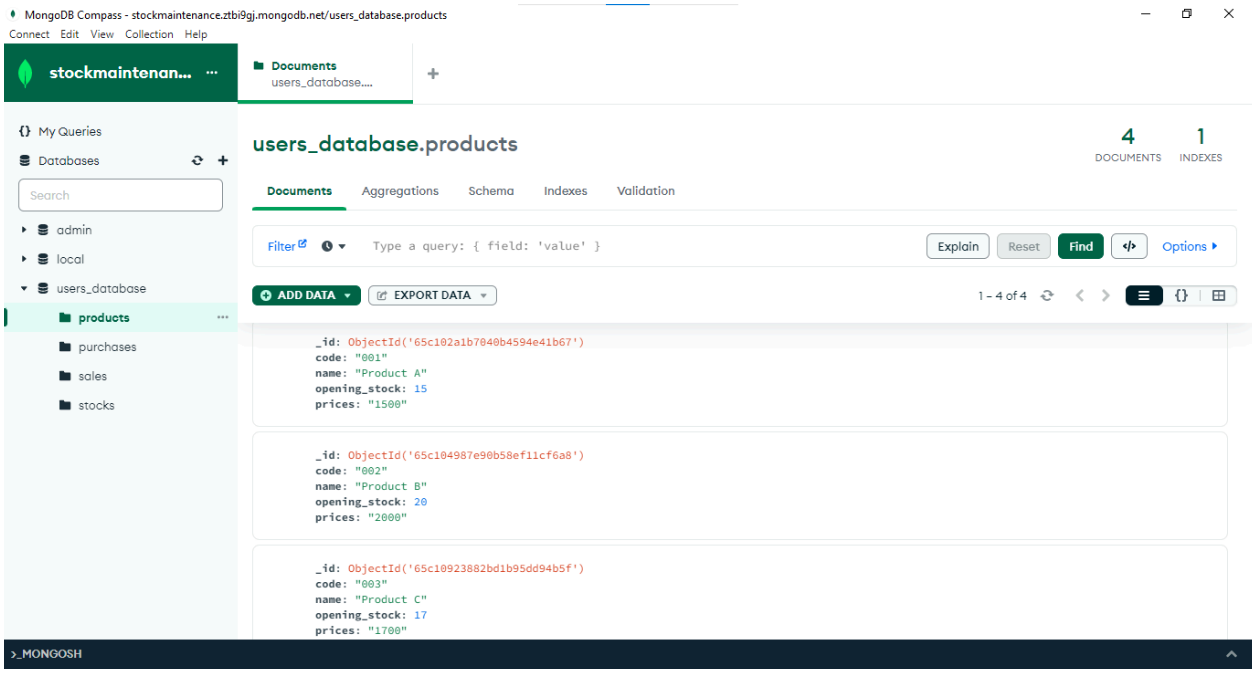
Stock Details

Product ID	Opening Stock	Purchase Stock	Sales Stock	Current Stock
004	18	0	0	18

Database: MongoDB Collections



Collection	Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
products	20.48 kB	4	92.00 B	1	36.86 kB
purchases	20.48 kB	4	87.00 B	1	36.86 kB
sales	20.48 kB	1	115.00 B	1	20.48 kB
stocks	20.48 kB	1	119.00 B	1	20.48 kB



```
_id: ObjectId('65c102a1b7040b4594e41b67')
code: "001"
name: "Product A"
opening_stock: 15
prices: "1500"

_id: ObjectId('65c104987e90b58ef11cf6a8')
code: "002"
name: "Product B"
opening_stock: 20
prices: "2000"

_id: ObjectId('65c10923882bd1b95dd94b5f')
code: "003"
name: "Product C"
opening_stock: 17
prices: "1700"
```

6. Conclusion:

The Stock Maintenance System successfully fulfills its objectives, offering an efficient and organized solution for businesses to manage their stock data. The use of Flask, MongoDB, and other web technologies ensures scalability and adaptability for future enhancements.