Faculty of Engineering and Technology

Computer Science Department

DATA BASE SYSTEMS COMP333

Second Semester 2024/2025

# BioLine Medical Company

**Prepared by:**

**Mohammed Salem – 1203022**

**Sabtiah Asad – 1221960**

**Instructor: Dr. Hanaa Qasrawi**

**Sec: 4**

**June 10th , 2025**

## Overview

BioLine Company is a comprehensive enterprise web application built using Flask and MySQL. It manages multiple company branches, departments, employees, suppliers, customers, and transactions. The system is designed to streamline administrative tasks, improve data visibility, and enable real-time reporting. It supports centralized CRUD operations and an advanced business intelligence dashboard for monitoring sales, inventory, and staff productivity.

## Table Of Contents

## Table Of Figures

## System Architecture

The BioLine Company application follows a modular, layered architecture that ensures scalability, maintainability, and a clean separation of concerns. It consists of the following primary components:

### 1. Frontend Layer (Presentation Layer)

- Technologies: HTML5, Jinja2 templates, CSS3, and vanilla JavaScript.
- Templates are modular and extend from a shared base.html, making them consistent.
- Interfaces are user-friendly and include modal-based forms, buttons, and dashboards for managing records.
- Responsive design is achieved through flexible layout structures and media queries in styles.css, ensuring compatibility across desktops, tablets, and mobile devices.

### 2. Backend Layer (Application Logic)

- Built with Flask, a lightweight Python web framework.
- Manages user authentication, session handling, route definitions, and server logic.
- Route functions handle CRUD operations, data validation, and dynamic rendering of pages.
- Special logic is implemented for:
  - Inventory updates during purchase and sale transactions.
  - Revenue and cost calculations during sales and purchases.
  - Business intelligence reporting (aggregated analytic and summaries).
- Includes a singleton-style database connector (Database-connection class) to efficiently reuse connections and reduce overhead.

### 3. Database Layer (Data Management)

- Utilizes MySQL.
- The schema is fully normalized, promoting data integrity and reducing redundancy.
- Includes 17+ well-structured tables such as Company, Department, Employee, Product, Customer, Supplier, and transaction tables like Sale and Purchase.
- Foreign keys and triggers ensure relational integrity and automatic updates:
  - Triggers maintain accurate EmployeeCount in departments and NumberOfEmployees in companies when staff are added or removed.
- Lookup/reference tables like PaymentMethod, CustomerType, and SupplierType ensure consistency and flexibility.

## 4. Security and Configuration

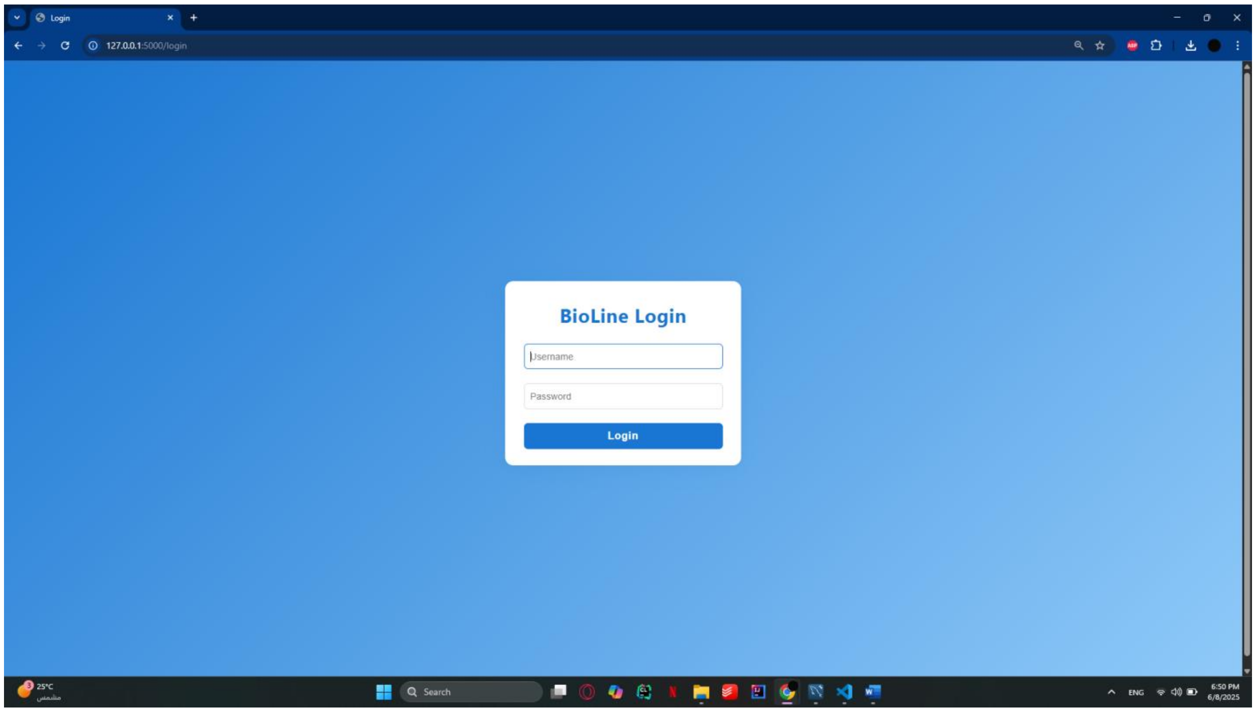
- Session validation for all routes.
- Future Upgrade: Password hashing (currently hardcoded for demo purposes).
- Config: Environment-specific settings (e.g., settings.json) to decouple credentials from code.

## 5. Routing and Template Engine

- Flask’s built-in Jinja2 engine is used to dynamically inject content into HTML templates.
- URL routing is handled by Flask decorators like @app.route(), mapping each page to a specific function.
- Data is passed to templates using Python dictionaries, ensuring smooth client-server communication.

## Authentication

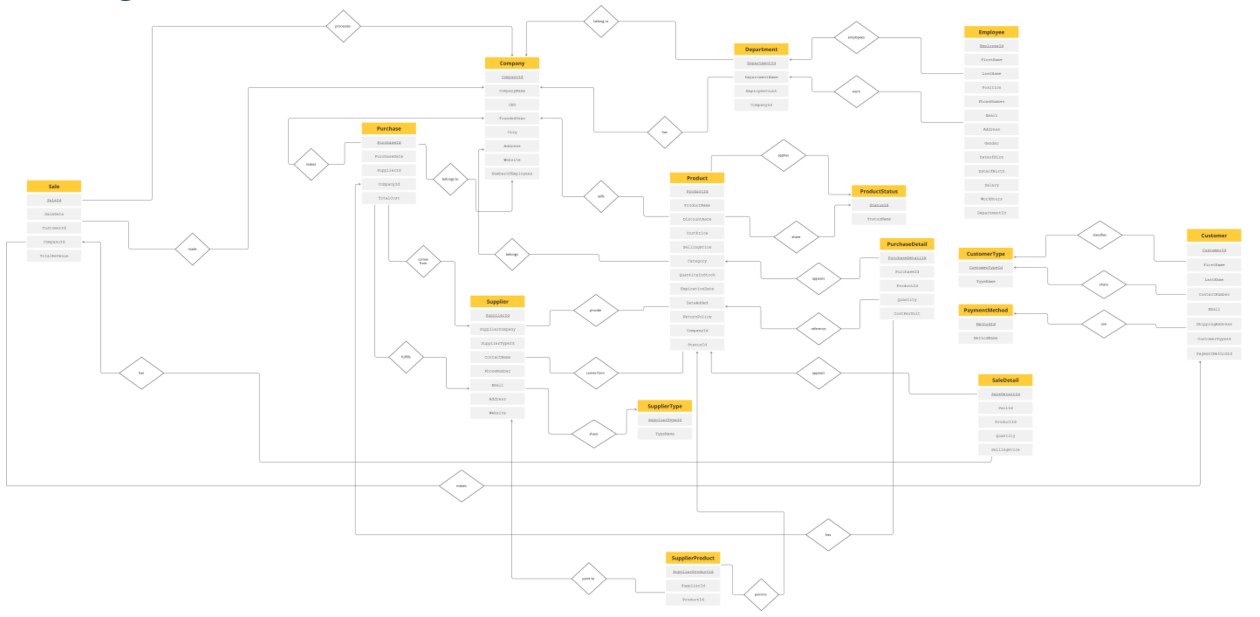
The system uses a basic session-based login page. Only admins can access the system (username: admin, password: 1234). Once authenticated, users are redirected to the dashboard, and session variables are used to restrict or allow access to other routes. The logout route clears the session and redirects the user to the login page.



## Database Schema

The MySQL database schema, BioLineCompany, consists of normalized tables for Company, Department, Employee, Product, Customer, Supplier, Purchase, Sale, and their respective details. Lookup tables are used for categories like PaymentMethod, CustomerType, ProductStatus, and SupplierType to ensure consistency and avoid redundancy. The schema is highly scalable and supports referential integrity through foreign key constraints.

## ER Diagram



## Data Normalization

### 1. First Normal Form (1NF):

- All tables have atomic values, and each table has a primary key.
- In BioLineCompany, all tables like Employee, Product, Customer, and Supplier use atomic columns such as FirstName, LastName, Salary, etc., and primary keys are defined such as CompanyId and ProductId.

### 2. Second Normal Form (2NF):

- Meets 1NF and that non-key attributes fully depend on the primary key.
- This form is especially relevant in tables with composite keys, such as:
- PurchaseDetail (PurchaseId, ProductId)
- SaleDetail (SaleId, ProductId)
- These linking tables ensure that each attribute (like Quantity, CostPerUnit, SellingPrice) depends on the full combination of keys, not just part of it.

### 3. Third Normal Form (3NF):

- Meets 2NF and ensures that non-key attributes are not transitively dependent on the primary key.
- Lookup tables: PaymentMethod, CustomerType, SupplierType, and ProductStatus store reference data to avoid redundancy.
- Employee references DepartmentId only — and not directly CompanyId, which it accesses transitively via the Department.

- Product depends on CompanyId and StatusId, which are foreign keys to normalized lookup tables.
- Customer stores CustomerTypeId and PaymentMethodId — removing descriptive types into separate reference tables.

| **Normalized Entity** | **Description** |
|---|---|
| Company | Core table, referenced by many others. No redundancy in branches. |
| Department | CompanyId FK + DepartmentName uniqueness per company (UNIQUE KEY (CompanyId, DepartmentName) |
| Employee | References DepartmentId; no repeated company info in Employee. |
| Product | Includes StatusId, which is a FK to ProductStatus (lookup table). |
| Customer | Separated CustomerTypeId and PaymentMethodId into own tables. |
| Supplier | Uses SupplierTypeId (local/international) from lookup table. |
| PurchaseDetail | Junction table between Purchase and Product; enforces composite logic. |
| SaleDetail | Junction table between Sale and Product; ensures per-product tracking. |
| SupplierProduct | Resolves many-to-many between Supplier and Product — a classic 3NF example. |

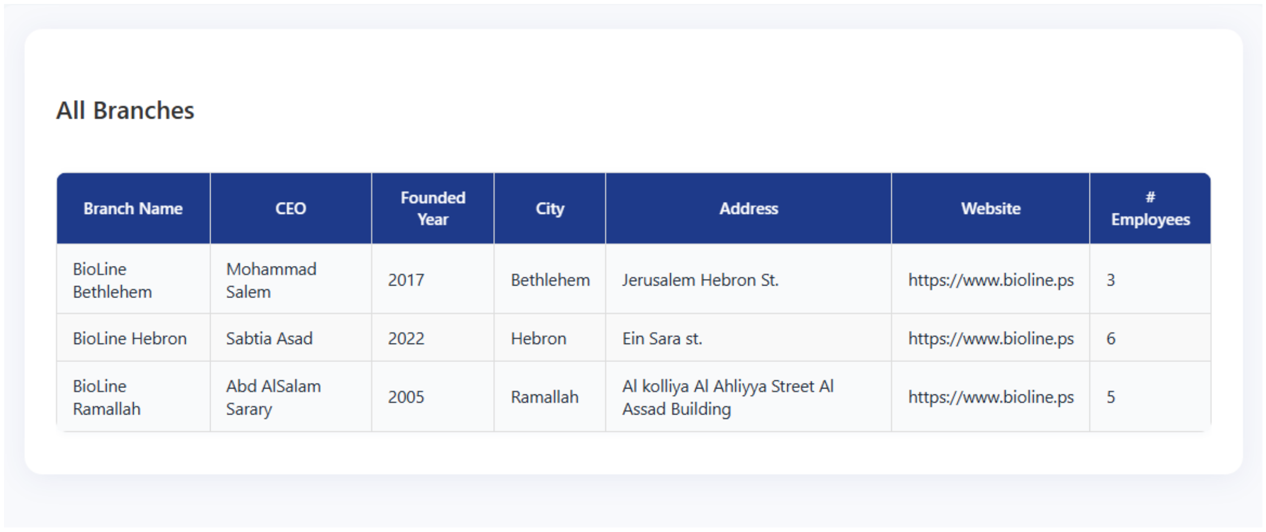
*Table 1:Normalized Entity*

## Triggers

The project includes MySQL triggers to maintain automatic consistency:

- When an employee is added or deleted, the EmployeeCount in the Department table updates automatically.
- The Company's NumberOfEmployees is dynamically recalculated as a sum of employees across all departments.

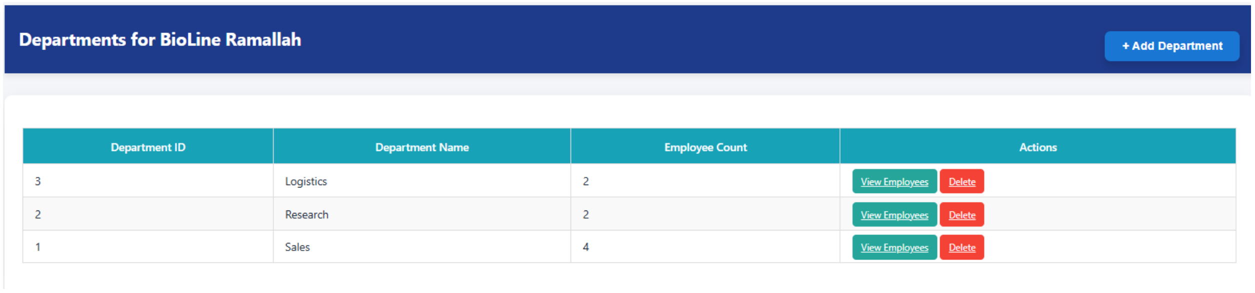
These triggers ensure data consistency without requiring manual updates from the application logic.



All Branches

| Branch Name | CEO | Founded Year | City | Address | Website | # Employees |
|---|---|---|---|---|---|---|
| BioLine Bethlehem | Mohammad Salem | 2017 | Bethlehem | Jerusalem Hebron St. | https://www.bioline.ps | 3 |
| BioLine Hebron | Sabtia Asad | 2022 | Hebron | Ein Sara st. | https://www.bioline.ps | 6 |
| BioLine Ramallah | Abd AlSalam Sarary | 2005 | Ramallah | Al kolliya Al Ahliyya Street Al Assad Building | https://www.bioline.ps | 5 |

*Figure 1- Branches Query*



Departments for BioLine Ramallah

| Department ID | Department Name | Employee Count | Actions |
|---|---|---|---|
| 3 | Logistics | 2 | View Employees Delete |
| 2 | Research | 2 | View Employees Delete |
| 1 | Sales | 4 | View Employees Delete |

*Figure 2 Department Query*

## Branches Page

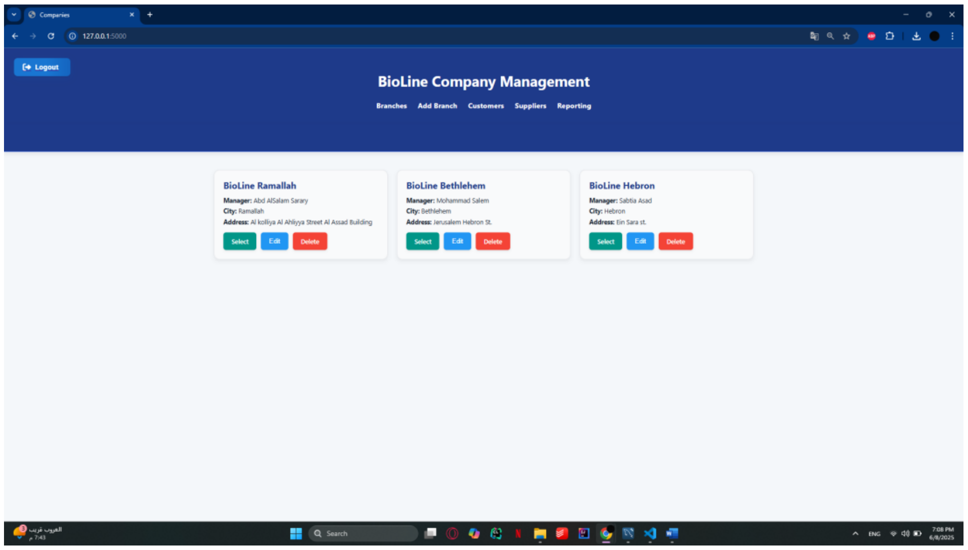
The Branches page (index.html) serves as the central hub for managing and accessing each company's branch within the BioLineCompany system. It provides an intuitive user interface for viewing, selecting, and navigating into specific company branches to manage related operations like departments, employees, sales, purchases, products, and reporting.

## Functionalities Provided

1. **List All Active Companies (Branches)**
   - Upon page load, all companies from the Company table that have IsActive = 1 are fetched and displayed.
   - Each row includes essential data such as:
     - Company Name

        - CEO
        - Founded Year
        - City
        - Number of Employees
        - Website

2. **Branch Control Panel**
    - For each company, a “View Dashboard” button is available.
    - Clicking it redirects the user to that company’s dashboard (/branch/<CompanyId>), where branch-specific modules (departments, employees, etc.) are accessible.
3. **Add New Branch**
    - A prominent “+ Add Branch” button triggers a modal form (or links to add_company.html).
    - The user can input new company details including name, CEO, city, address, and website.
    - On submission, data is inserted into the Company table and NumberOfEmployees is initialized to zero.
4. **Edit or Delete Branch**
    - For each listed branch, options are available to Edit or Delete.
    - Edit opens a modal pre-filled with the current data allowing updates.
    - Delete triggers a warning prompt; upon confirmation, the selected branch is deleted, which also cascades deletions to:
        - Departments
        - Employees
        - Products
        - Sales and Purchases
    - This cascade is enforced via ON DELETE CASCADE in SQL schema.

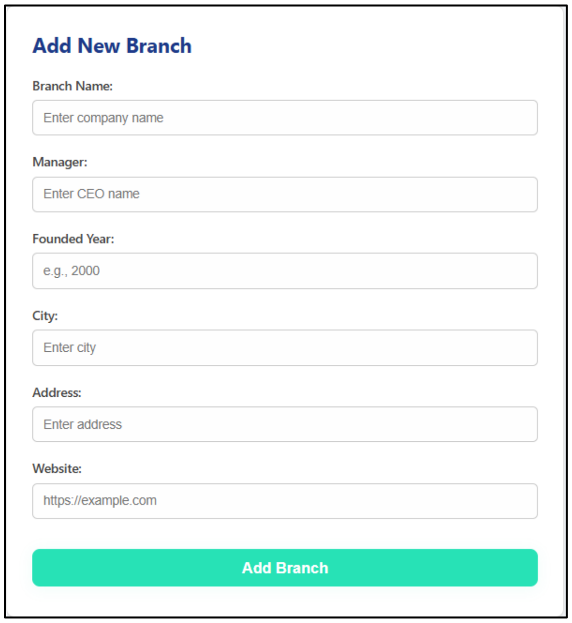


**BioLine Ramallah**
**Manager:** Abd AlSalam Sarary
**City:** Ramallah
**Address:** Al kolliya Al Ahliyya Street Al Assad Building

**BioLine Bethlehem**
**Manager:** Mohammad Salem
**City:** Bethlehem
**Address:** Jerusalem Hebron St.

**BioLine Hebron**
**Manager:** Sabtia Asad
**City:** Hebron
**Address:** Ein Sara st.

*Figure 3 Main Page*


Add New Branch

Branch Name:

Enter company name

Manager:

Enter CEO name

Founded Year:

e.g., 2000

City:

Enter city

Address:

Enter address

Website:

https://example.com

Add Branch

*Figure 4 Add Branch Form*

## Branch Dashboard Page

Route: /branch/<int:company_id>

## Description:

The Branch Dashboard page serves as the entry point for all internal operations within a specific company (or branch) selected from the homepage. It displays a simple welcome message and a sidebar navigation menu for managing core modules of that branch.

## Functional Overview

When a user selects a branch like "BioLine Ramallah", they are redirected to a page specifically tied to that branch's CompanyId. Here's what the page enables:

### *Welcome Section*

- Displays the company name (e.g., **BioLine Ramallah - Management**) in the header.
- Shows a short instructional message encouraging users to navigate using the sidebar.

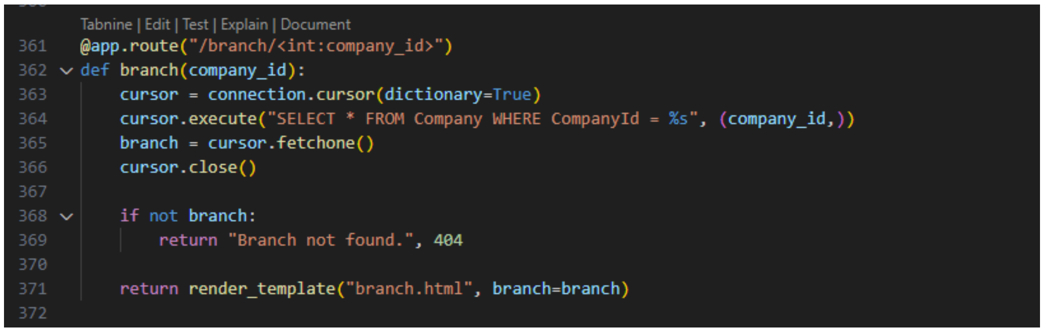
### *Sidebar Navigation*

Located on the left, this sidebar gives direct access to:

- **Dashboard**: Reloads the branch home (this page).

- **Departments**: Leads to /departments/<CompanyId> where all departments under this company can be managed.
- **Products**: Leads to /products/<CompanyId> to view, add, edit, or remove products in this branch.
- **Purchases**: Leads to /purchase/<CompanyId> where purchase records and suppliers are handled.
- **Sales**: Leads to /sales/<CompanyId> for managing customer transactions.

The logic for this page is implemented in the Flask backend (app.py) like so:



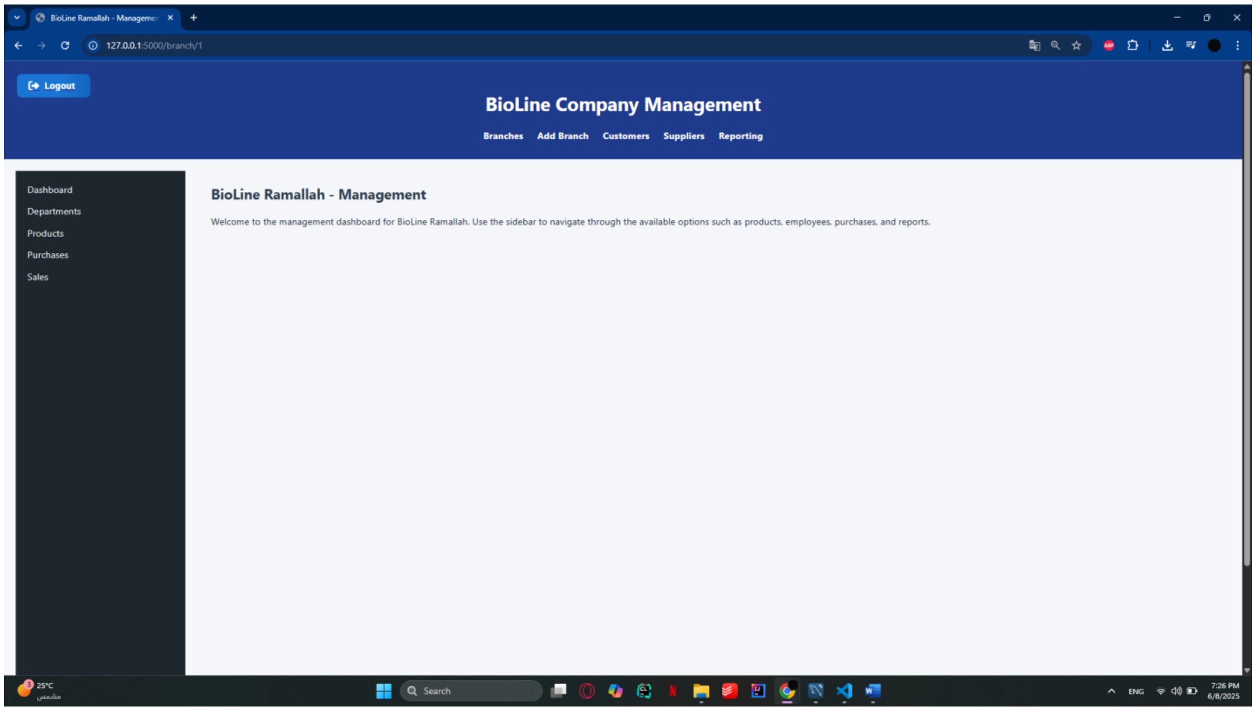
```python
@app.route("/branch/<int:company_id>")
def branch(company_id):
    cursor = connection.cursor(dictionary=True)
    cursor.execute("SELECT * FROM Company WHERE CompanyId = %s", (company_id,))
    branch = cursor.fetchone()
    cursor.close()

    if not branch:
        return "Branch not found.", 404

    return render_template("branch.html", branch=branch)
```

*Figure 5 app.py <Branches>*

- ￼ This route dynamically loads the selected company's data from the database.
- ￼ If the company does not exist or is deactivated, a 404 message is returned.
- ￼ The branch name is passed into the branch.html template for display and dynamic navigation.



BioLine Company Management

BioLine Ramallah - Management

Welcome to the management dashboard for BioLine Ramallah. Use the sidebar to navigate through the available options such as products, employees, purchases, and reports.

*Figure 6 Branch Page*

## Departments Page

Route: /departments/<int:company_id>

### Description:

The Departments Page is a core administrative view that allows users to manage the internal organizational structure of a selected company (branch). It lists all departments for a given company and provides the ability to add, delete, and explore the employees within each department.

### Functionalities

#### 1. List Departments

- Displays all departments that belong to the selected CompanyId.
- The following columns are shown in the department table:
    - **Department ID** – The unique identifier of the department.
    - **Department Name** – The descriptive name (e.g., "Sales", "Research").
    - **Employee Count** – A live value showing how many employees are currently assigned to the department.

    - **Actions** – Interactive buttons for further actions.

**2. Add Department**

- Add Department button in the top right opens a form (modal or separate page).
- This allows the admin to create a new department by entering its name.
- Upon submission:
    - The department is added to the Department table with the associated CompanyId.
    - The EmployeeCount is initialized to 0.
    - The department will appear in the list immediately after creation.

**3. Delete Department**

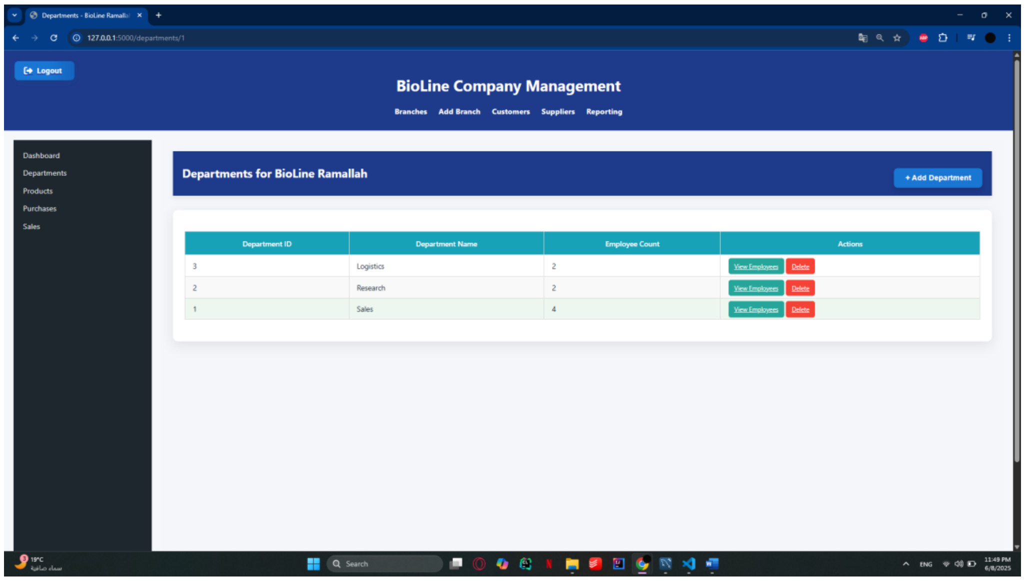
- Each department row includes a Delete button.
- Clicking this triggers a backend action to remove the department from the database.
- On deletion:
    - The department is removed from the Department table.
    - Cascading restrictions or ON DELETE RESTRICT in the schema prevent deletion if employees still exist in the department (to preserve integrity).

**4. View Employees**

- Each department row includes a View Employees button.
- Clicking this redirects the user to /employees/<department_id>, where all employees under that department are listed.
- This provides quick access to staff management and related personnel details.

**5. Employee Count Tracking**

- The Employee Count column shows how many employees are assigned to that department.
- This value is **automatically managed** by MySQL triggers:
    - When an employee is added, the EmployeeCount is incremented.
    - When an employee is deleted, the EmployeeCount is decremented.



Departments for BioLine Ramallah

| Department ID | Department Name | Employee Count | Actions |
|---|---|---|---|
| 3 | Logistics | 2 | View Employees Delete |
| 2 | Research | 2 | View Employees Delete |
| 1 | Sales | 4 | View Employees Delete |

*Figure 7 Departments Page*

## Employees Page

Route: /employees/<int:department_id>

### Description:

The Employees Page is designed to manage the personnel assigned to a specific department within a branch (company). It provides a clean, table-based layout listing all employees for that department, with options to add, edit, or delete employees.

### Functionalities

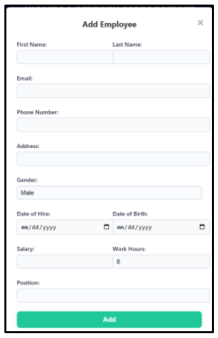
#### 1. View Employees

- The table displays key employee information, including:
  - Employee ID
  - First Name
  - Last Name
  - Email
  - Phone Number
- Only employees from the selected department are shown.

#### 2. Add Employee

- Add Employee button ( opens a modal form).
- Admin fills in:

- First and Last Name
- Email
- Phone Number
- Gender
- Position
- Date of Hire & Date of Birth
- Salary
- Address
- Work hours



Add Employee

| | |
|---|---|
| First Name: | Last Name: |
| Email: | |
| Phone Number: | |
| Address: | |
| Gender: Male | |
| Date of Hire: mm/dd/yyyy | Date of Birth: mm/dd/yyyy |
| Salary: | Work Hours: 8 |
| Position: | |

*Figure 8:add employee form*

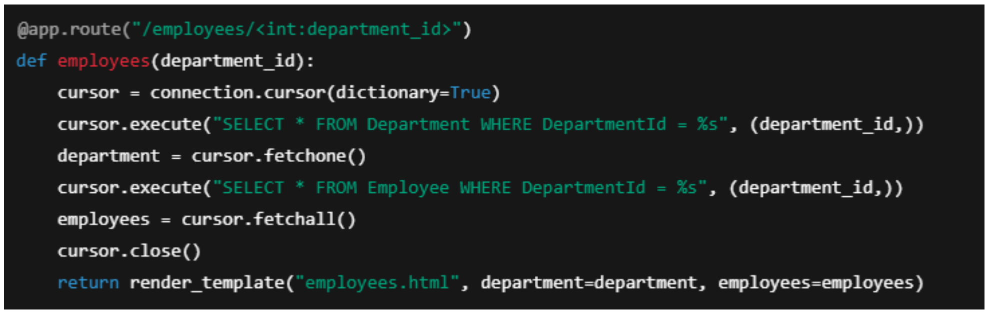
- On form submission:
  - Data is inserted into the Employee table with a DepartmentId FK.
  - The EmployeeCount in the Department table is automatically incremented using a MySQL trigger.
  - The NumberOfEmployees in the related Company is also updated using a cascading trigger.

**3. Edit Employee**

- Each row includes an Edit button.
- This pre-fills a form with the employee's existing data for updates.
- After editing, changes are saved to the database.
- The page refreshes to reflect the updated record immediately.

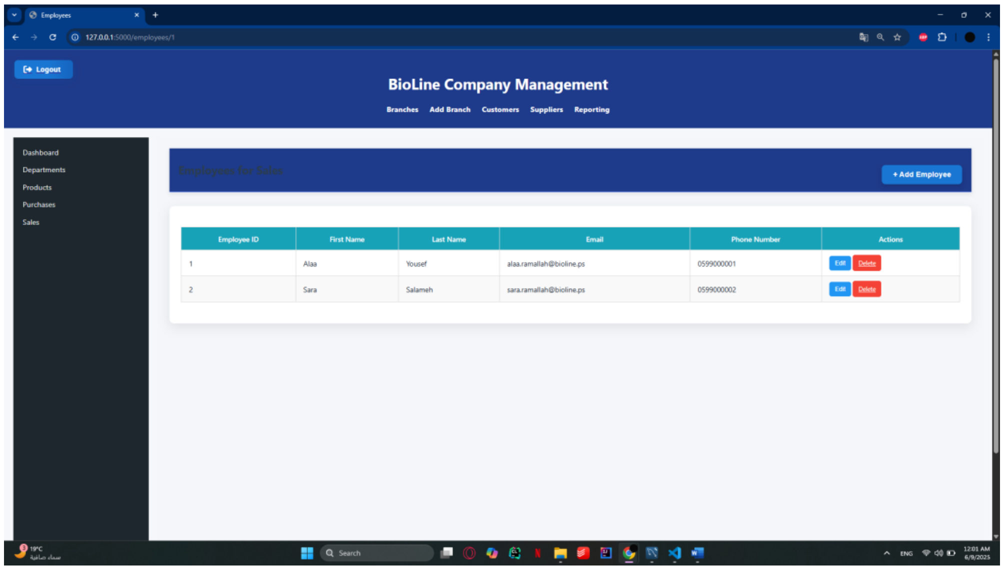
**4. Delete Employee**

- Each row includes a Delete button.
- Clicking it removes the employee from the database.
- The EmployeeCount for the department is automatically decremented by a trigger.
- If that employee was the last in a department, the count will reflect as 0.



```
@app.route("/employees/<int:department_id>")
def employees(department_id):
    cursor = connection.cursor(dictionary=True)
    cursor.execute("SELECT * FROM Department WHERE DepartmentId = %s", (department_id,))
    department = cursor.fetchone()
    cursor.execute("SELECT * FROM Employee WHERE DepartmentId = %s", (department_id,))
    employees = cursor.fetchall()
    cursor.close()
    return render_template("employees.html", department=department, employees=employees)
```

- Loads the department's name and ID.
- Fetches all active employees tied to that department.



| Employee ID | First Name | Last Name | Email | Phone Number | Actions |
|---|---|---|---|---|---|
| 1 | Alaa | Yousef | alaa.ramallah@bioline.ps | 0599000001 | Edit Delete |
| 2 | Sara | Salameh | sara.ramallah@bioline.ps | 0599000002 | Edit Delete |

*Figure 9: employee page*

## Products Page

Route: /products/<int:company_id>

### Description:

The Products Page allows company branches to manage their inventory of products. Each branch has its own set of products, and this page displays a full listing of all inventory items with detailed information and management actions.

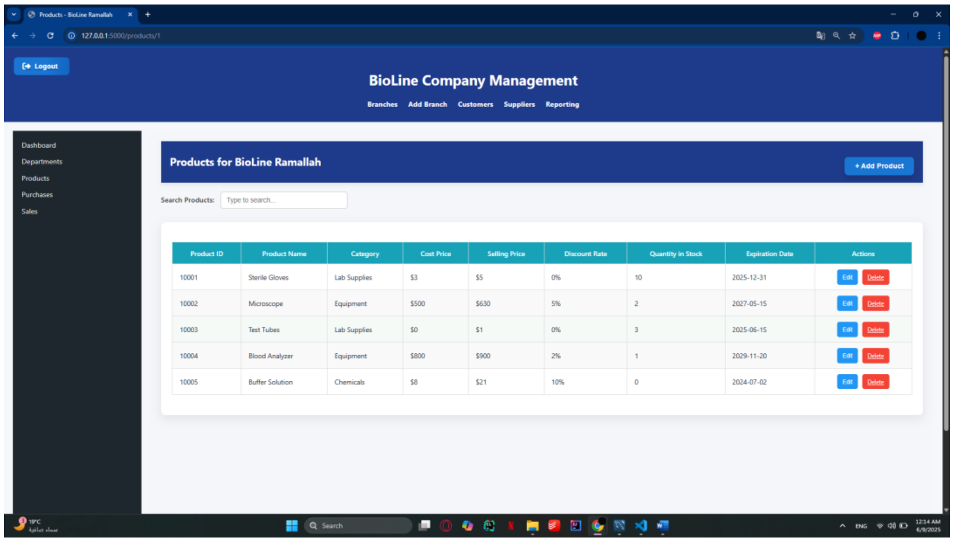
### Functionalities

#### 1. List Products

The product table includes the following columns:

- **Product ID** – Unique identifier
- **Product Name** – Descriptive item name
- **Category** – Type of product (e.g., Equipment, Chemicals)
- **Cost Price** – Purchase price of the product
- **Selling Price** – Price offered to customers
- **Discount Rate** – Any applied discount in percentage
- **Quantity in Stock** – How many units are currently available
- **Expiration Date** – Shown for perishable or chemical products

- **Actions** – Includes buttons to Edit or Delete the product



Products for BioLine Ramallah

Search Products:

| Product ID | Product Name | Category | Cost Price | Selling Price | Discount Rate | Quantity in Stock | Expiration Date | Actions |
|---|---|---|---|---|---|---|---|---|
| 10001 | Sterile Gloves | Lab Supplies | $3 | $5 | 0% | 10 | 2025-12-31 | Edit Delete |
| 10002 | Microscope | Equipment | $500 | $630 | 5% | 2 | 2027-05-15 | Edit Delete |
| 10003 | Test Tubes | Lab Supplies | $0 | $1 | 0% | 3 | 2025-06-15 | Edit Delete |
| 10004 | Blood Analyzer | Equipment | $800 | $900 | 2% | 1 | 2029-11-20 | Edit Delete |
| 10005 | Buffer Solution | Chemicals | $8 | $21 | 10% | 0 | 2024-07-02 | Edit Delete |

*Figure 10:Products Page*

### 2. Add Product

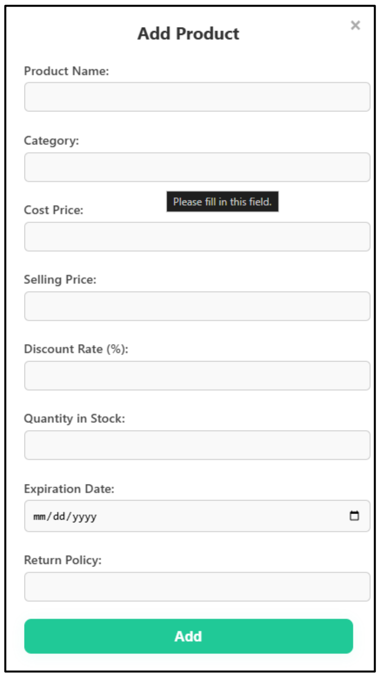


*Figure 11 Add Product Form*

- Add Product button opens a form to create a new product.
- The admin inputs:
  - Product name, category, cost and selling prices
  - Optional discount rate
  - Quantity available
  - Expiration date
  - Return policy (optional)
- On submission:
  - Product is added to the Product table.
  - CompanyId is set to link the product with the current branch.
  - The StatusId is defaulted to "Available" using the ProductStatus lookup table.
- Products appear in the list instantly after creation.

### 3. Edit Product

- Each product has an Edit button.

- Clicking opens a modal or form with pre-filled data.
- Admin can update:
    - Name, category, prices, quantity, expiration date, discount, return policy
- After updating, the database is modified, and the table reloads with the latest data.

**4. Delete Product**

- The **Delete** button removes the product from the system.
- Once deleted:
    - The product is removed from the Product table.
    - Foreign key constraints (e.g., in SaleDetail, PurchaseDetail) ensure products involved in transactions cannot be deleted unless detached first.

## Purchase Page

Route: /purchase/<int:company_id>

### Description:

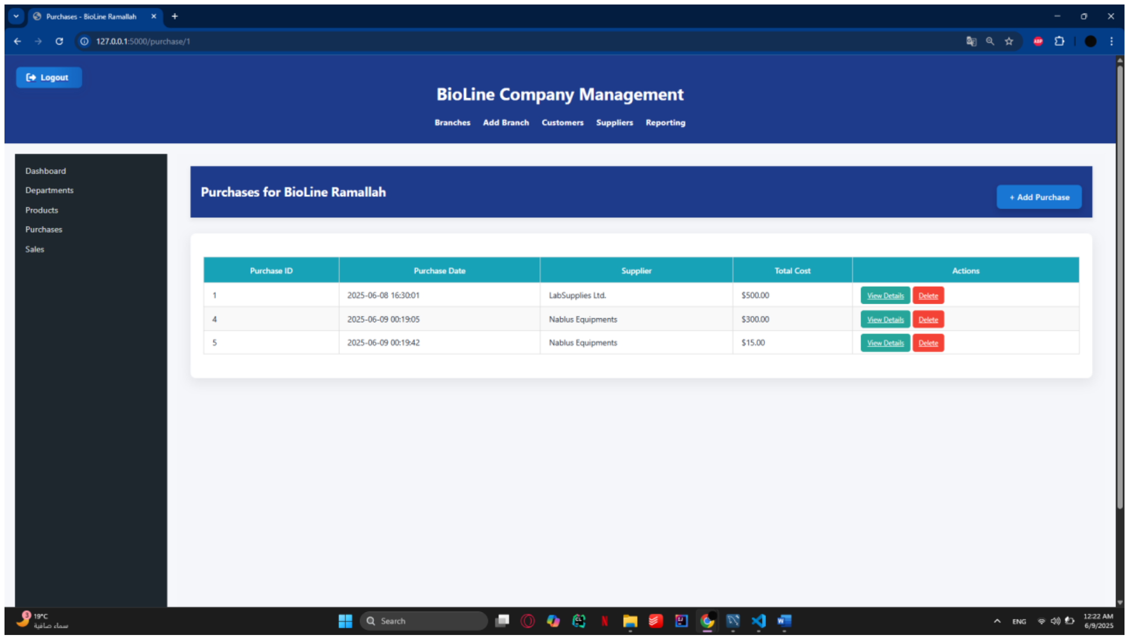
The Purchases Page is designed to track the procurement of products for a selected branch (company). It allows admins to record purchases, link them to suppliers, update inventory levels, and analyze spending. This page is critical for supply chain visibility and stock replenishment.

### Functionalities

**1. List Purchases**

Displays all purchases made by the current company. Each row includes:

- **Purchase ID** – Auto-incremented primary key
- **Purchase Date** – Timestamp of the transaction
- **Supplier** – Name of the supplier from whom items were purchased
- **Total Cost** – Automatically calculated based on product cost × quantity
- **Actions** – Includes buttons to view itemized details or delete the purchase



BioLine Company Management

Purchases for BioLine Ramallah

| Purchase ID | Purchase Date | Supplier | Total Cost | Actions |
|---|---|---|---|---|
| 1 | 2025-06-08 16:30:01 | LabSupplies Ltd. | $500.00 | View Details Delete |
| 4 | 2025-06-09 00:19:05 | Nablus Equipments | $300.00 | View Details Delete |
| 5 | 2025-06-09 00:19:42 | Nablus Equipments | $15.00 | View Details Delete |

*Figure 12:Prurchase page*

### 2. Add Purchase

- Clicking the  Add Purchase button opens a detailed form.
- Admin selects:
  - Supplier
  - One or more products from the company's inventory
  - Quantity and cost per unit for each selected product

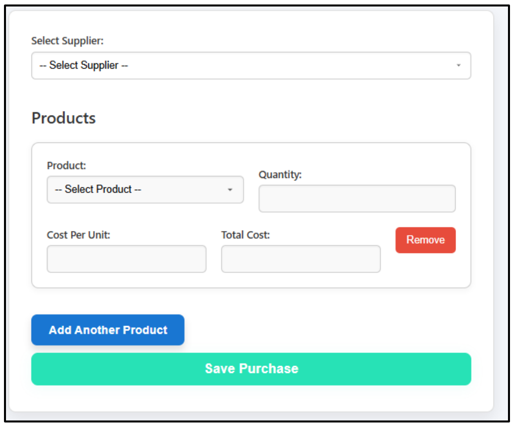


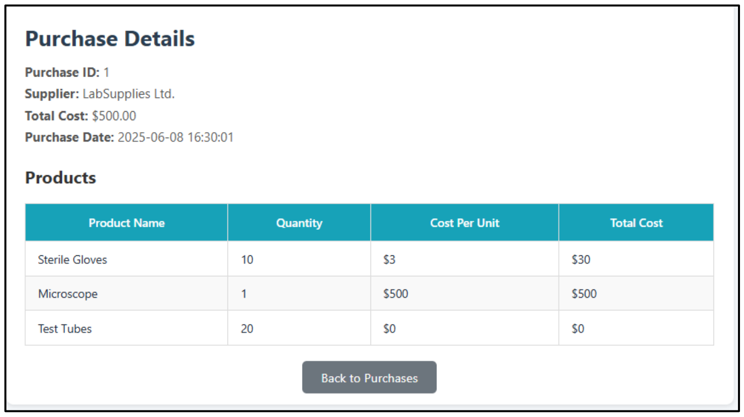
*Figure 13: add purchase form*

- When submitted:
  - A record is inserted into the Purchase table.
  - Individual entries for each product are added to PurchaseDetail.
  - Product QuantityInStock values are automatically increased.
  - Total cost is calculated and saved into the main purchase record.

### 3. View Details

- The **View Details** button navigates to /purchase_details/<purchase_id>.

- A new page shows:
    - Supplier name
    - Purchase timestamp
    - Each purchased product
    - Quantity and cost per unit
    - Total cost per product
    - Overall total cost
- Enables financial transparency and purchase validation.



Purchase Details

**Purchase ID:** 1
**Supplier:** LabSupplies Ltd.
**Total Cost:** $500.00
**Purchase Date:** 2025-06-08 16:30:01

Products

| Product Name | Quantity | Cost Per Unit | Total Cost |
|---|---|---|---|
| Sterile Gloves | 10 | $3 | $30 |
| Microscope | 1 | $500 | $500 |
| Test Tubes | 20 | $0 | $0 |

*Figure 14: purchase Details Page*

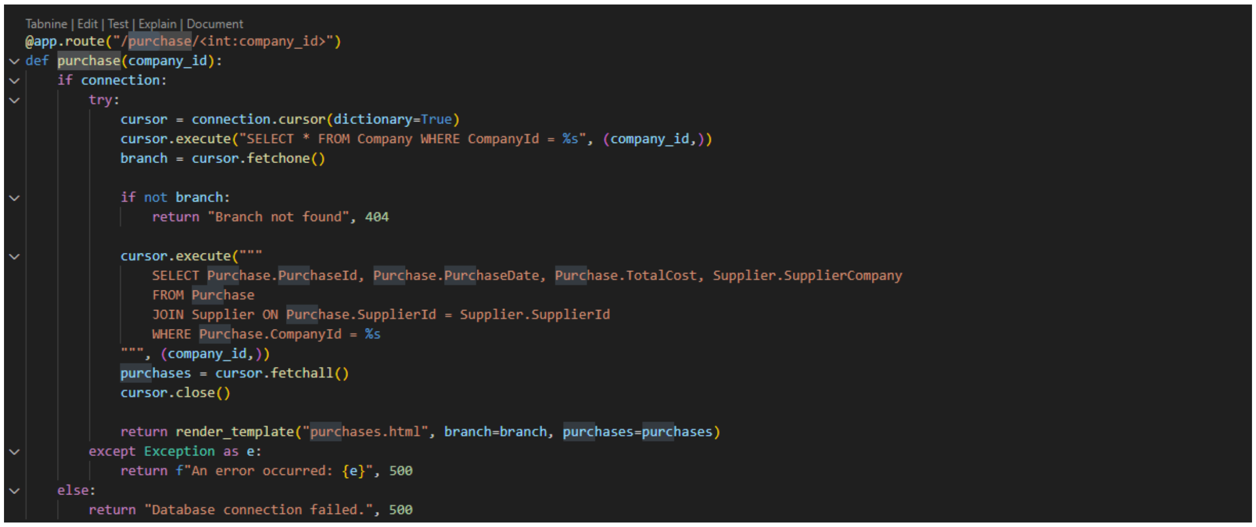
 **4. Delete Purchase**

- Clicking the **Delete** button:
    - Deletes all related rows from PurchaseDetail
    - Then deletes the main Purchase record
- This keeps the database clean and prevents orphaned details
- Inventory may need adjustment if rollback logic is required (future enhancement)

**Inventory Management Logic**

- Quantity tracking is automated during transactions:
    - A purchase adds to QuantityInStock
    - A sale subtracts from it
- Products with QuantityInStock = 0 or those expiring within 30 days are flagged on the reporting dashboard.
- This ensures real-time stock visibility and reduces the risk of selling out-of-stock items.

- The system ensures that purchasing **adds stock**:
    - UPDATE Product SET QuantityInStock = QuantityInStock + quantity
- This is done for each item as part of the /add_purchase backend route.
- Ensures accurate, live inventory tracking per company.



```python
@app.route("/purchase/<int:company_id>")
def purchase(company_id):
    if connection:
        try:
            cursor = connection.cursor(dictionary=True)
            cursor.execute("SELECT * FROM Company WHERE CompanyId = %s", (company_id,))
            branch = cursor.fetchone()

            if not branch:
                return "Branch not found", 404

            cursor.execute("""
                SELECT Purchase.PurchaseId, Purchase.PurchaseDate, Purchase.TotalCost, Supplier.SupplierCompany
                FROM Purchase
                JOIN Supplier ON Purchase.SupplierId = Supplier.SupplierId
                WHERE Purchase.CompanyId = %s
            """, (company_id,))
            purchases = cursor.fetchall()
            cursor.close()

            return render_template("purchases.html", branch=branch, purchases=purchases)
        except Exception as e:
            return f"An error occurred: {e}", 500
    else:
        return "Database connection failed.", 500
```

*Figure 15:Purchase Code*

## Sales Page

Route: /sales/<int:company_id>

### Description:

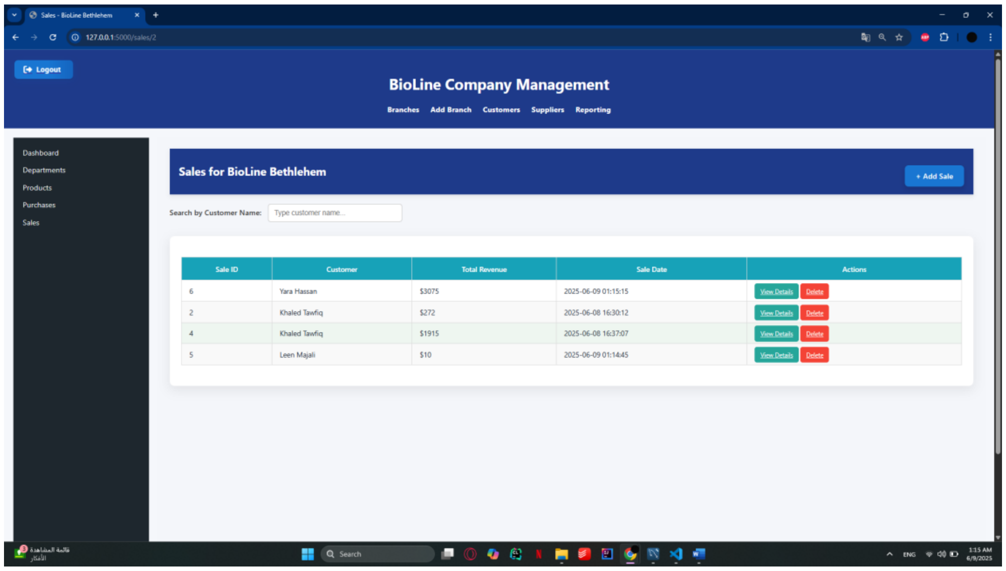
The Sales Page provides an interface to manage and track all sales transactions conducted by a specific company branch. Each sale is linked to a customer, and multiple products can be sold in a single transaction. This page is critical for analyzing revenue, managing stock, and ensuring order traceability.

### Functionalities

#### 1. List Sales

Each row in the table includes:

- **Sale ID** – Unique auto-incremented identifier
- **Sale Date** – Timestamp of when the sale was made
- **Customer Name** – Name of the customer (joined from the Customer table)
- **Total Revenue** – Automatically calculated total of the transaction
- **Actions** – Includes buttons for:
    - **View Details** – View all items sold in that sale
    - **Delete** – Remove the sale and adjust stock accordingly



BioLine Company Management

Branches Add Branch Customers Suppliers Reporting

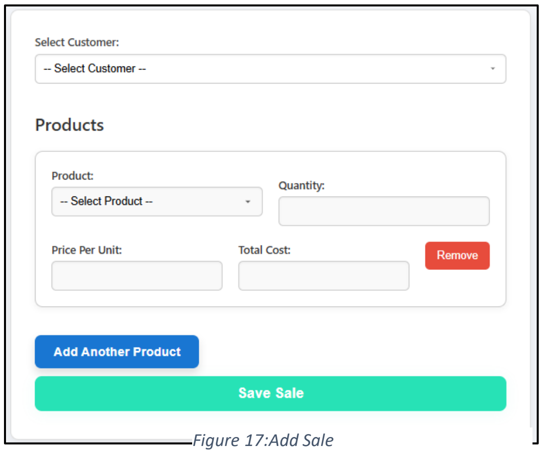
Sales for BioLine Bethlehem

+ Add Sale

Search by Customer Name: Type customer name...

| Sale ID | Customer | Total Revenue | Sale Date | Actions |
|---|---|---|---|---|
| 6 | Yara Hassan | $3075 | 2025-06-09 01:15:15 | View Details Delete |
| 2 | Khaled Tawfiq | $272 | 2025-06-08 16:30:12 | View Details Delete |
| 4 | Khaled Tawfiq | $1915 | 2025-06-08 16:37:07 | View Details Delete |
| 5 | Leen Majali | $10 | 2025-06-09 01:14:45 | View Details Delete |

*Figure 16:Sales Page*


Select Customer:
-- Select Customer --

Products

Product: -- Select Product --
Quantity:
Price Per Unit:
Total Cost:

*Figure 17:Add Sale*

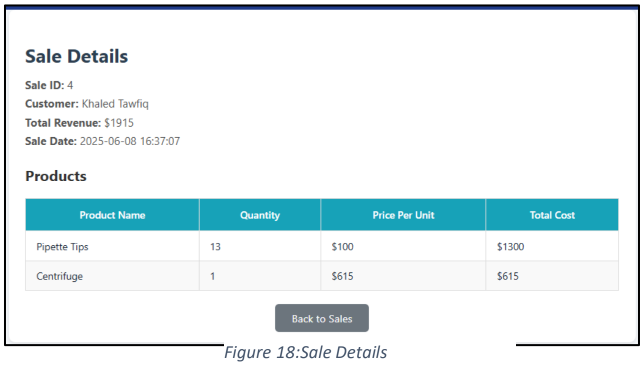
## 2. Add Sale

- Clicking  Add Sale opens a comprehensive form.
- Admin selects:
    - A **customer** from the list
    - One or more **products**
    - Quantity for each product (must not exceed current stock)
- For each product, the price and total cost are shown.
- On submission:
    - The sale is inserted into the Sale table.
    - Each item is saved in SaleDetail.
    - **Product stock is automatically reduced** based on quantity sold.
    - Total revenue is calculated and saved.

## 3. View Details

- Opens /sale_details/<sale_id> page.
- Displays:
    - Sale ID, Date, Customer, and Branch

    - A list of sold products with quantity, price, and total per item
    - The overall transaction revenue
- Enables transaction auditing and transparency



## Sale Details

**Sale ID:** 4
**Customer:** Khaled Tawfiq
**Total Revenue:** $1915
**Sale Date:** 2025-06-08 16:37:07

**Products**

| Product Name | Quantity | Price Per Unit | Total Cost |
|---|---|---|---|
| Pipette Tips | 13 | $100 | $1300 |
| Centrifuge | 1 | $615 | $615 |

*Figure 18:Sale Details*

### 4. Delete Sale

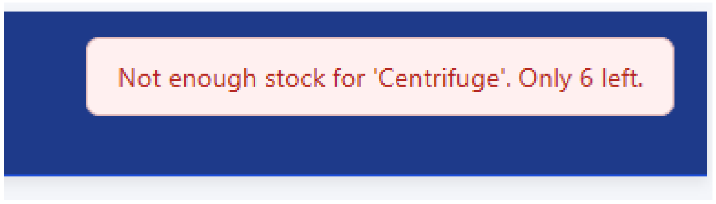
- Clicking **Delete** removes both:
    - The record from Sale
    - Associated items in SaleDetail
- Optionally, stock rollback can be implemented to re-add quantities to inventory.

## Inventory Deduction

- During the sale:

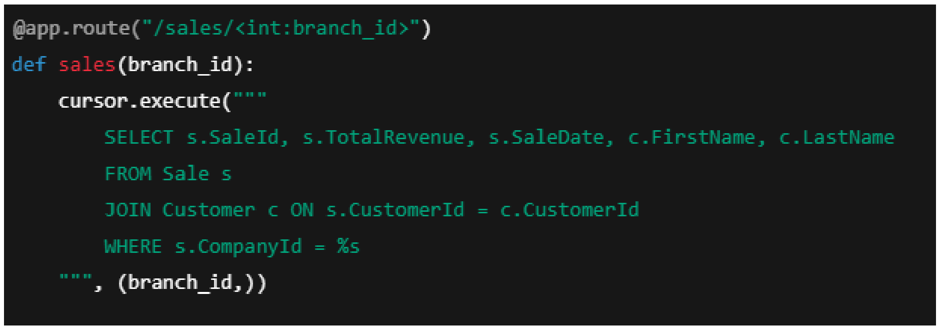
UPDATE Product SET QuantityInStock = QuantityInStock - <quantity>

- Inventory deduction is validated to **never go below zero**.
- An error is shown if a requested quantity exceeds the available stock.



Not enough stock for 'Centrifuge'. Only 6 left.

*Figure 19:sale more than quantity in stock*



```
@app.route("/sales/<int:branch_id>")
def sales(branch_id):
    cursor.execute("""
        SELECT s.SaleId, s.TotalRevenue, s.SaleDate, c.FirstName, c.LastName
        FROM Sale s
        JOIN Customer c ON s.CustomerId = c.CustomerId
        WHERE s.CompanyId = %s
    """, (branch_id,))
```

*Figure 20 Sale Code*

- Sales are filtered by CompanyId
- Total revenue is shown for each transaction
- The customer name is constructed by joining with the Customer table

### Purpose & Business Impact

- Provides complete visibility into branch-level sales activity
- Automatically adjusts product stock levels
- Helps assess customer behavior and purchasing patterns
- Acts as the foundation for revenue analytics, dashboards, and reports
- Enables detailed transaction histories and supports audit trails

## Customers Page

Route: `/customers`

### Description:

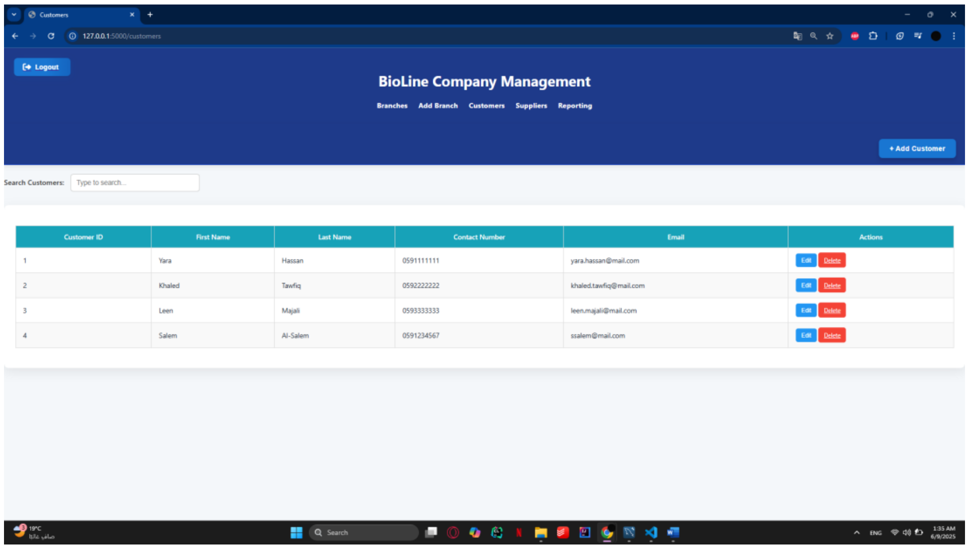
The Customer Page provides a centralized interface to manage all customer records used across company branches. Admins can view, add, update, or delete customer profiles, which are later used in sales transactions. This module ensures that customer information is clean, organized, and easily searchable.

**Functionalities**

**1. List Customers**

Displays a full list of customers in a structured table format, including:

- **Customer ID** – Primary key identifier
- **First Name / Last Name**
- **Email**
- **Phone Number**
- **Shipping Address**
- **Customer Type** – Linked via FK (Retail / Wholesale)
- **Payment Method** – Linked via FK (Cash / Credit / Bank Transfer)
- **Actions** – Includes **Edit** and **Delete** buttons



| Customer ID | First Name | Last Name | Contact Number | Email | Actions |
|---|---|---|---|---|---|
| 1 | Yara | Hassan | 0591111111 | yara.hassan@mail.com | Edit Delete |
| 2 | Khaled | Tawfiq | 0592222222 | khaled.tawfiq@mail.com | Edit Delete |
| 3 | Leen | Majali | 0593333333 | leen.majali@mail.com | Edit Delete |
| 4 | Salem | Al-Salem | 0591234567 | ssalem@mail.com | Edit Delete |

*Figure 21:Customers Page*

### 2. Add Customer

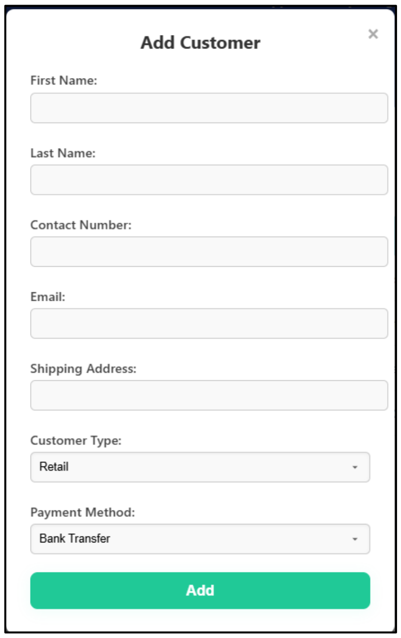


*Figure 22:Add Customer Form*

- Clicking the Add Customer button opens a form to create a new customer.
- Required fields:
  - First and Last Name
  - Contact Number
  - Email
  - Address
  - Dropdown for:
    - **Customer Type** (from the CustomerType table)
    - **Payment Method** (from the PaymentMethod table)
- On submission:
  - The customer is inserted into the Customer table.
  - Foreign key references ensure valid CustomerTypeId and PaymentMethodId.

### 3. Edit Customer

- The Edit button on each row pre-fills the form with existing values.
- Admin can update any field.
- Upon submission:

  - Changes are saved to the database using an UPDATE statement.
  - Data immediately refreshes in the UI.

**4. Delete Customer**

- Clicking **Delete** removes the customer from the system.
- The customer can no longer be used in sales, and all associated foreign key constraints are respected (cannot delete if sales exist unless cascade is allowed).

**5. Search/Filter Customers**

- A search bar is available to filter the customer list.
- Users can type part of a name, email, or contact number to narrow results.
- Improves efficiency in locating a specific customer in large datasets.

## Suppliers Page

### Route: `/suppliers`

### Description:

The Suppliers Page is a dedicated interface for managing all external suppliers who provide goods and inventory to BioLineCompany branches. Suppliers are critical to procurement operations, and this page ensures their data is well maintained and easily accessible.

It allows administrators to add, edit, delete, and search supplier records, each of which is linked to purchases made within the system.

### Functionalities

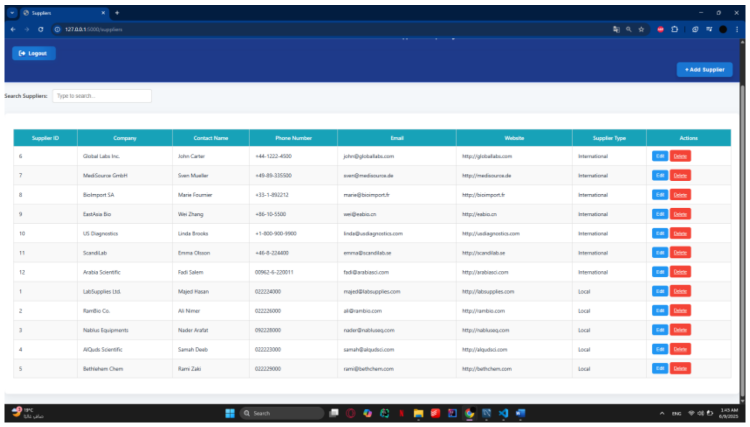
**1. List Suppliers**

The main table displays:

- **Supplier ID** – Unique identifier
- **Supplier Company** – Name of the company
- **Contact Name** – Person responsible for communication
- **Phone Number**
- **Email**
- **Address**
- **Website**
- **Supplier Type** – Lookup value (e.g., Local or International)

- **Actions** – Buttons for Edit and Delete

The supplier list is paginated and styled for clarity and accessibility.



| Supplier ID | Company | Contact Name | Phone Number | Email | Website | Supplier Type | Actions |
|---|---|---|---|---|---|---|---|
| 6 | Global Labs Inc. | John Carter | +44-1222-4500 | john@globallabs.com | http://globallabs.com | International | Edit Delete |
| 7 | MediSource GmbH | Sven Mueller | +49-89-335500 | sven@medisource.de | http://medisource.de | International | Edit Delete |
| 8 | BioImport SA | Marie Fournier | +33-1-892212 | marie@bioimport.fr | http://bioimport.fr | International | Edit Delete |
| 9 | EastAsia Bio | Wei Zhang | +86-10-5500 | wei@eabio.cn | http://eabio.cn | International | Edit Delete |
| 10 | US Diagnostics | Linda Brooks | +1-800-900-9900 | linda@usdiagnostics.com | http://usdiagnostics.com | International | Edit Delete |
| 11 | ScandiLab | Emma Olsson | +46-8-224400 | emma@scandilab.se | http://scandilab.se | International | Edit Delete |
| 12 | Arabia Scientific | Fadi Salem | 00962-6-220011 | fadi@arabiasci.com | http://arabiasci.com | International | Edit Delete |
| 1 | LabSupplies Ltd. | Majed Hasan | 022224000 | majed@labsupplies.com | http://labsupplies.com | Local | Edit Delete |
| 2 | RamBio Co. | Ali Nimer | 022226000 | ali@rambio.com | http://rambio.com | Local | Edit Delete |
| 3 | Nablus Equipments | Nader Arafat | 092228000 | nader@nabluseq.com | http://nabluseq.com | Local | Edit Delete |
| 4 | AlQuds Scientific | Samah Deeb | 022223000 | samah@alqudsci.com | http://alqudsci.com | Local | Edit Delete |
| 5 | Bethlehem Chem | Rami Zaki | 022229000 | rami@bethchem.com | http://bethchem.com | Local | Edit Delete |

*Figure 23 Suppliers Page*

## 2. Add Supplier

- Clicking the Add Supplier button opens a form to register a new supplier.
- Required fields:
  - Company name
  - Contact name
  - Phone, email, address
  - Website (optional)
  - **Supplier Type** – selected from the dropdown populated by the SupplierType table (Local or International)
- On submission:
  - The supplier is added to the Supplier table.
  - Foreign key integrity is enforced by linking to SupplierTypeId.

## 3. Edit Supplier

- Each supplier has an Edit button.
- When clicked, it opens a pre-filled form with the current supplier's data.
- Admin can update any field, including contact info or supplier type.
- On saving:
  - The changes are updated in the Supplier table.

    - The UI reloads with the latest data reflected.

**4. Delete Supplier**

- The **Delete** button removes a supplier from the system.
- If the supplier is referenced in existing purchases, deletion may be restricted based on foreign key constraints (or handled with ON DELETE RESTRICT).
- Ensures the database does not allow orphaned purchase records.

**5. Search Suppliers**

- A **search input** allows filtering the supplier list by:
    - Supplier company name
    - Contact name , City or other details

## Reporting & Business Intelligence System

**Routes:**

- /reporting – Main reporting dashboard
- /company_financials – Financial analytics per company
- /employees_by_dept – Employee explorer by company and department

### Description:

The Reporting system in BioLineCompany is designed to provide real-time business insights, drawn from sales, purchases, products, employees, and customers. These analytics help management make informed decisions across departments and branches.

It combines SQL-based data aggregation with dynamic HTML dashboards using Flask and Jinja2.

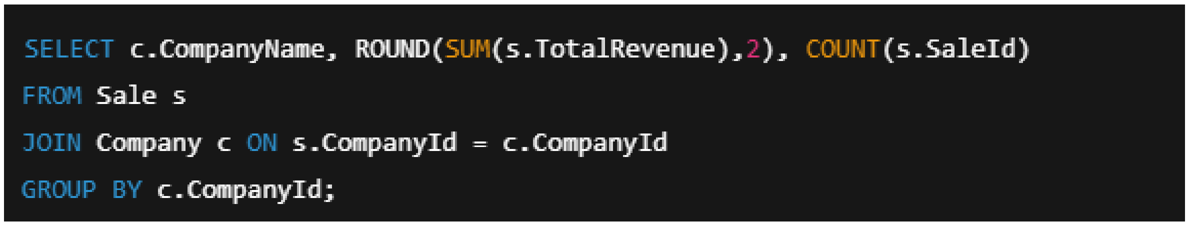
### Queries in the Reporting System

1. **Revenue by Company**
2. **Top-Selling Products**
3. **Employees per Company and Salary**
4. **Employees per Department (within Companies)**
5. **Detailed Employee List**
6. **Recent Hires**
7. **Inventory by Company**
8. **Inventory Value by Company**

9. **Expiring Products (within 30 days)**
10. **Out of Stock Products**
11. **Products by Category**
12. **Sales by Month**
13. **Revenue by Product**
14. **Recent Orders**
15. **Customer Spending Summary**
16. **Supplier Product Contributions**
17. **Supplier Purchases Summary**
18. **Company Financials (Revenue, Expenditure, Profit)**
19. **Employees by Department Tool (searchable)**
20. **Total number of employees in company and the total paid salary**
21. **List all employees**
22. **Average Price for Catogery**

## Detailed Explanation of Each Report

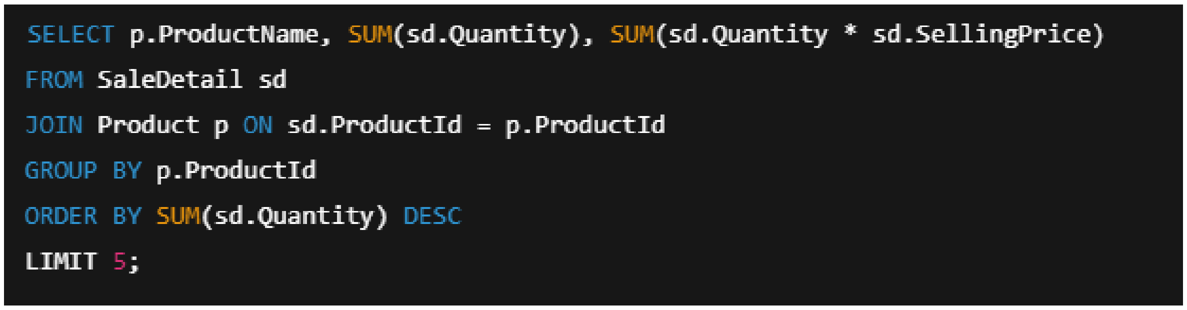
### 1. Revenue by Company



```sql
SELECT c.CompanyName, ROUND(SUM(s.TotalRevenue),2), COUNT(s.SaleId)
FROM Sale s
JOIN Company c ON s.CompanyId = c.CompanyId
GROUP BY c.CompanyId;
```

Calculates total revenue and number of sales per company branch. Useful for evaluating branch performance.

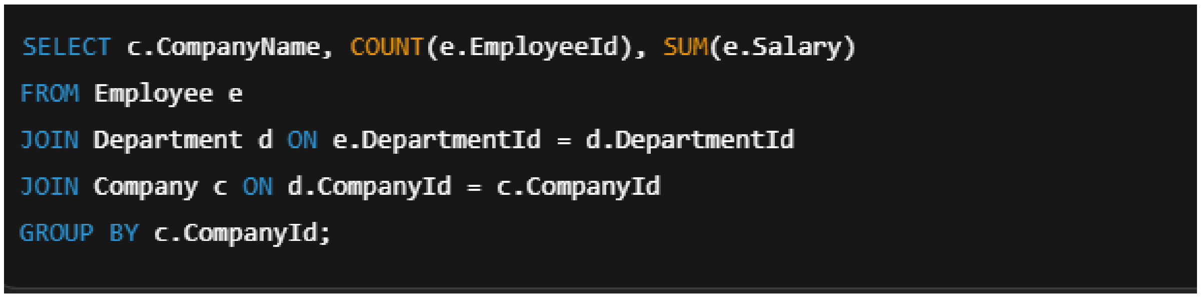
### 2. Top-Selling Products



```sql
SELECT p.ProductName, SUM(sd.Quantity), SUM(sd.Quantity * sd.SellingPrice)
FROM SaleDetail sd
JOIN Product p ON sd.ProductId = p.ProductId
GROUP BY p.ProductId
ORDER BY SUM(sd.Quantity) DESC
LIMIT 5;
```

Identifies the top5 products with the highest quantity sold. Informs product demand and marketing.

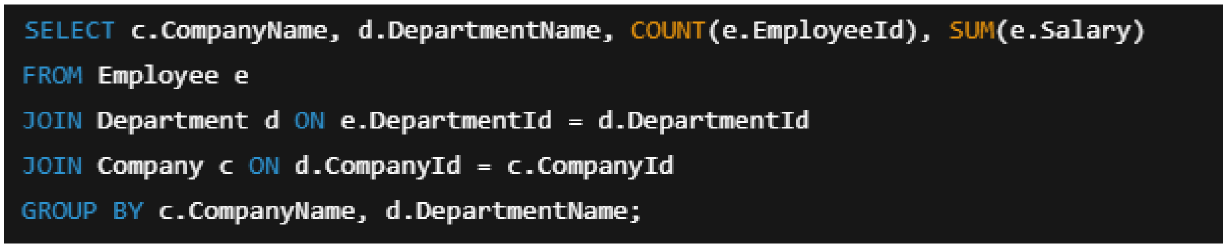
3. Employees per Company and Salary



```sql
SELECT c.CompanyName, COUNT(e.EmployeeId), SUM(e.Salary)
FROM Employee e
JOIN Department d ON e.DepartmentId = d.DepartmentId
JOIN Company c ON d.CompanyId = c.CompanyId
GROUP BY c.CompanyId;
```

Shows how many employees each company has and the total salary burden.

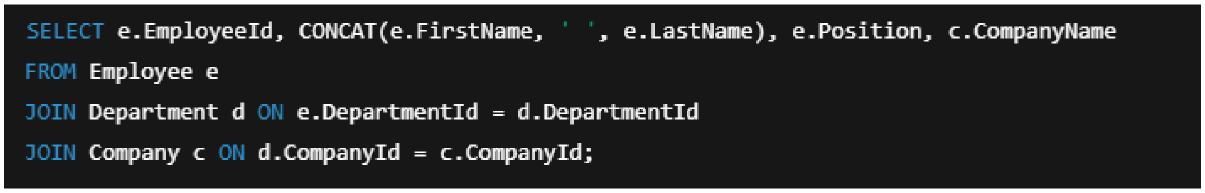
4. Employees per Department



```sql
SELECT c.CompanyName, d.DepartmentName, COUNT(e.EmployeeId), SUM(e.Salary)
FROM Employee e
JOIN Department d ON e.DepartmentId = d.DepartmentId
JOIN Company c ON d.CompanyId = c.CompanyId
GROUP BY c.CompanyName, d.DepartmentName;
```

Breaks down employees and salaries by department within each company.

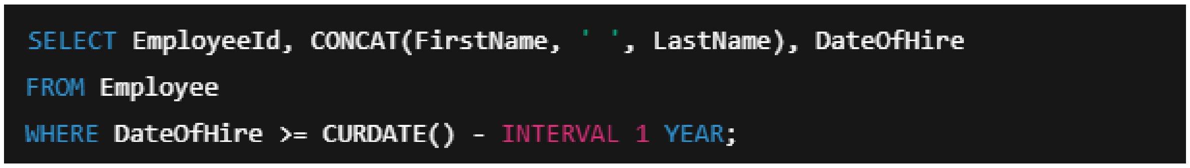
5. Detailed Employee List



```sql
SELECT e.EmployeeId, CONCAT(e.FirstName, ' ', e.LastName), e.Position, c.CompanyName
FROM Employee e
JOIN Department d ON e.DepartmentId = d.DepartmentId
JOIN Company c ON d.CompanyId = c.CompanyId;
```

Full list of all active employees with their position, email, and phone.

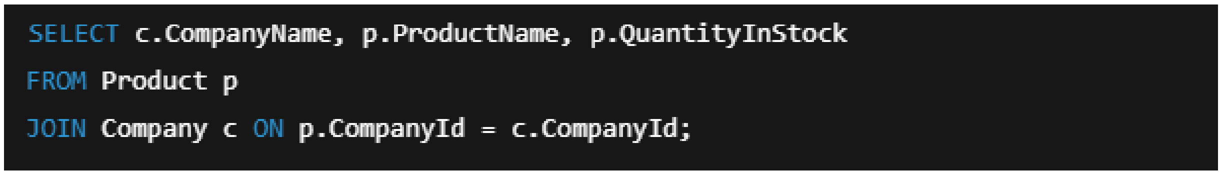
6. Recent Hires



```sql
SELECT EmployeeId, CONCAT(FirstName, ' ', LastName), DateOfHire
FROM Employee
WHERE DateOfHire >= CURDATE() - INTERVAL 1 YEAR;
```

Lists employees hired within the past year. Helps track workforce growth.

7. Inventory by Company



```sql
SELECT c.CompanyName, p.ProductName, p.QuantityInStock
FROM Product p
JOIN Company c ON p.CompanyId = c.CompanyId;
```

Lists each product and its available quantity per company.

8. Inventory Value by Company

```sql
SELECT c.CompanyName, SUM(p.QuantityInStock * p.CostPrice)
FROM Product p
JOIN Company c ON p.CompanyId = c.CompanyId
GROUP BY c.CompanyId;
```

Estimates the total monetary value of unsold inventory per company.

9. Expiring Products (next 30 days)

```sql
SELECT ProductId, ProductName, ExpirationDate
FROM Product
WHERE ExpirationDate BETWEEN CURDATE() AND CURDATE() + INTERVAL 30 DAY;
```

Highlights products nearing expiry. Useful for discount or clearance planning.

10. Out of Stock Products

```sql
SELECT ProductId, ProductName
FROM Product
WHERE QuantityInStock = 0;
```

Lists all unavailable products, indicating restocking is needed.

11. Products by Category

```sql
SELECT Category, COUNT(ProductId), AVG(SellingPrice)
FROM Product
GROUP BY Category;
```

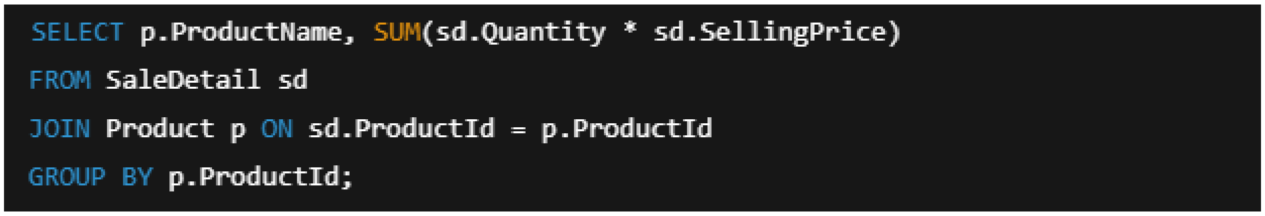
Shows how many products exist per category and their average price.

12. Sales by Month

```sql
SELECT DATE_FORMAT(SaleDate, '%Y-%m') AS Month, SUM(TotalRevenue)
FROM Sale
GROUP BY Month;
```

Visualizes monthly revenue trends over time.

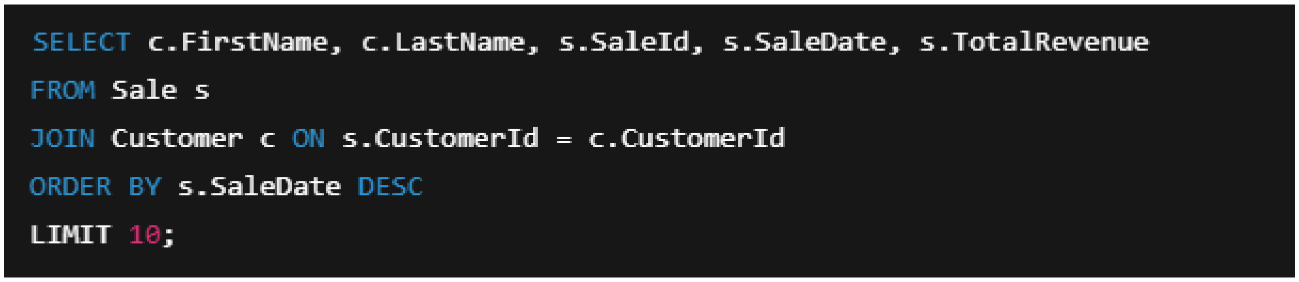
### 13. Revenue by Product



```sql
SELECT p.ProductName, SUM(sd.Quantity * sd.SellingPrice)
FROM SaleDetail sd
JOIN Product p ON sd.ProductId = p.ProductId
GROUP BY p.ProductId;
```

Shows total revenue earned from each product.

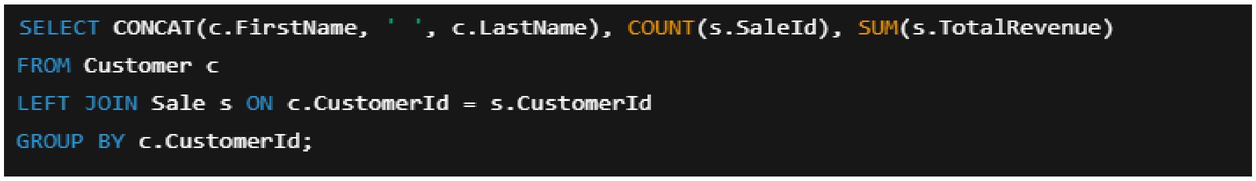
### 14. Recent Orders



```sql
SELECT c.FirstName, c.LastName, s.SaleId, s.SaleDate, s.TotalRevenue
FROM Sale s
JOIN Customer c ON s.CustomerId = c.CustomerId
ORDER BY s.SaleDate DESC
LIMIT 10;
```

Displays the 10 latest customer orders.

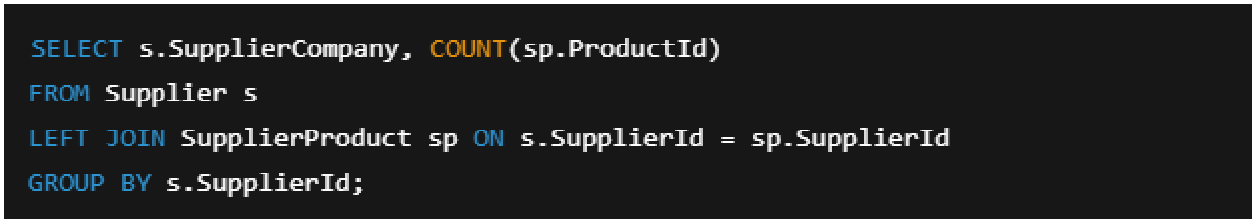
### 15. Customer Spending



```sql
SELECT CONCAT(c.FirstName, ' ', c.LastName), COUNT(s.SaleId), SUM(s.TotalRevenue)
FROM Customer c
LEFT JOIN Sale s ON c.CustomerId = s.CustomerId
GROUP BY c.CustomerId;
```

Ranks customers based on number of orders and total money spent.

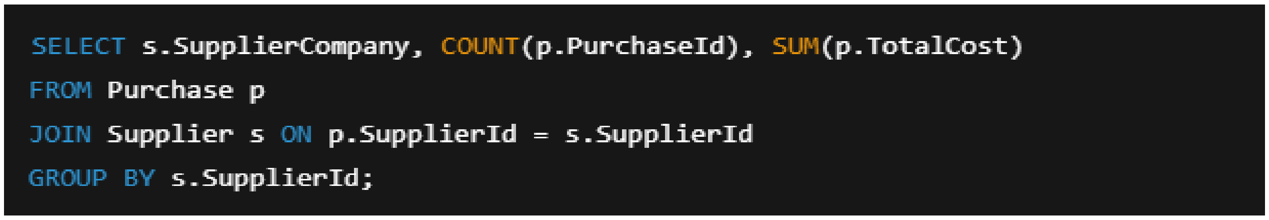
### 16. Supplier Product Counts



```sql
SELECT s.SupplierCompany, COUNT(sp.ProductId)
FROM Supplier s
LEFT JOIN SupplierProduct sp ON s.SupplierId = sp.SupplierId
GROUP BY s.SupplierId;
```

Tracks how many products each supplier is linked to.

### 17. Supplier Purchases Summary



```sql
SELECT s.SupplierCompany, COUNT(p.PurchaseId), SUM(p.TotalCost)
FROM Purchase p
JOIN Supplier s ON p.SupplierId = s.SupplierId
GROUP BY s.SupplierId;
```

Indicates how frequently and how much each supplier is used.

## Company Financials Page

### Route: /company_financials

This page shows:

- **Total Revenue** (from sales)
- **Total Expenditure** (from purchases)
- **Profit** (revenue - expenditure)
- **Low Stock Products** (≤5 units)
- **Employee Salary Costs** for the selected company

Filters:

- Company selection (via dropdown)
- Date range for revenue and expenditure

**Summary**: Provides a concise financial health check for any branch, including cost, profit, salaries, and inventory issues.

## Employees by Department Tool

### Route: /employees_by_dept

Features:

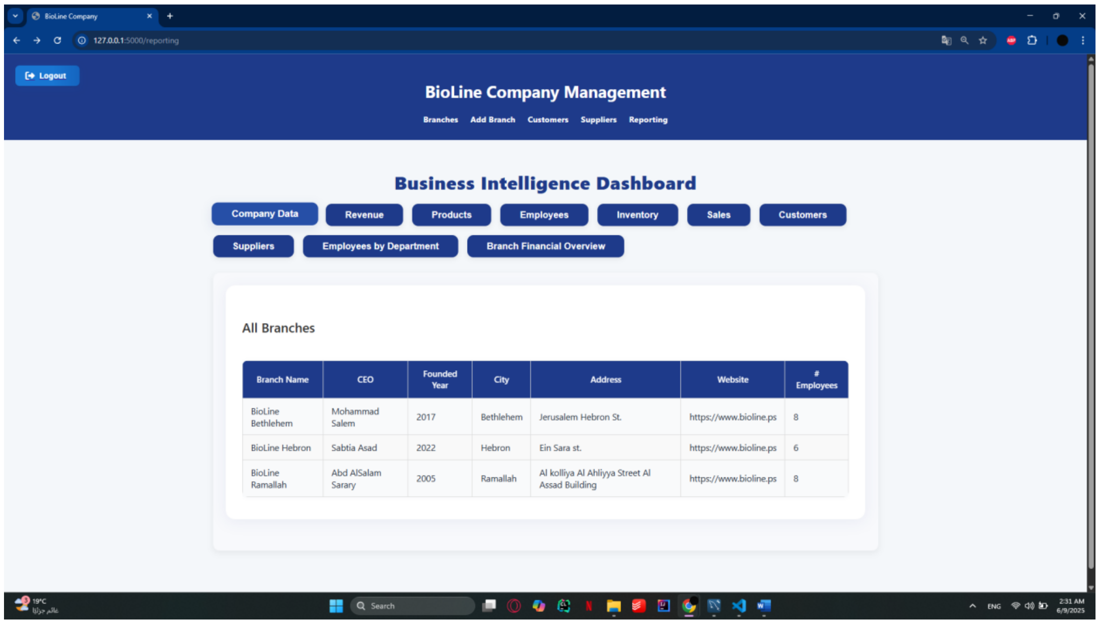
- Company dropdown to filter departments
- Department dropdown to view its employees
- Shows employee name, contact info, salary, birth/hire date

Summary: A dynamic lookup tool to explore employees in real-time by organizational hierarchy.

The reporting engine in BioLineCompany acts as the analytical brain of the system. It draws from normalized, real-time relational data to produce:

- Visual breakdowns of operations
- Metrics on sales, purchases, inventory, employees, and suppliers
- Tools for financial and HR planning
- Decision-making aids for executives

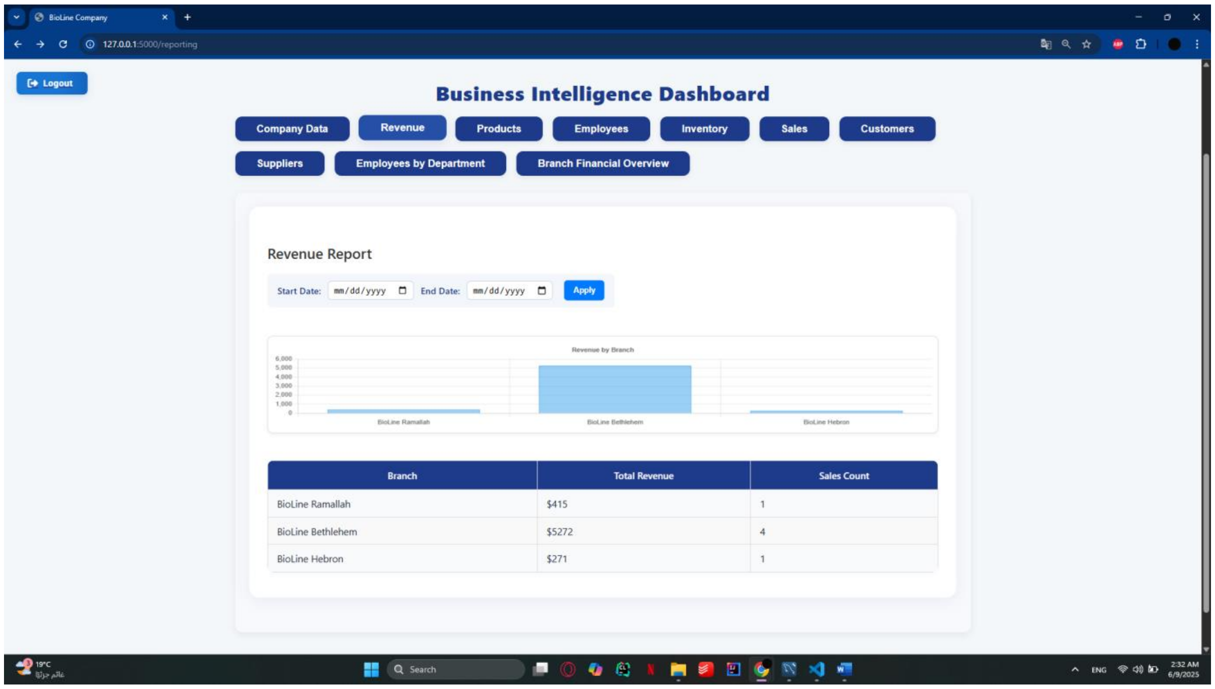
Screenshots Shows All Queries Results:



All Branches

| Branch Name | CEO | Founded Year | City | Address | Website | # Employees |
|---|---|---|---|---|---|---|
| BioLine Bethlehem | Mohammad Salem | 2017 | Bethlehem | Jerusalem Hebron St. | https://www.bioline.ps | 8 |
| BioLine Hebron | Sabtia Asad | 2022 | Hebron | Ein Sara st. | https://www.bioline.ps | 6 |
| BioLine Ramallah | Abd AlSalam Sarary | 2005 | Ramallah | Al kolliya Al Ahliyya Street Al Assad Building | https://www.bioline.ps | 8 |

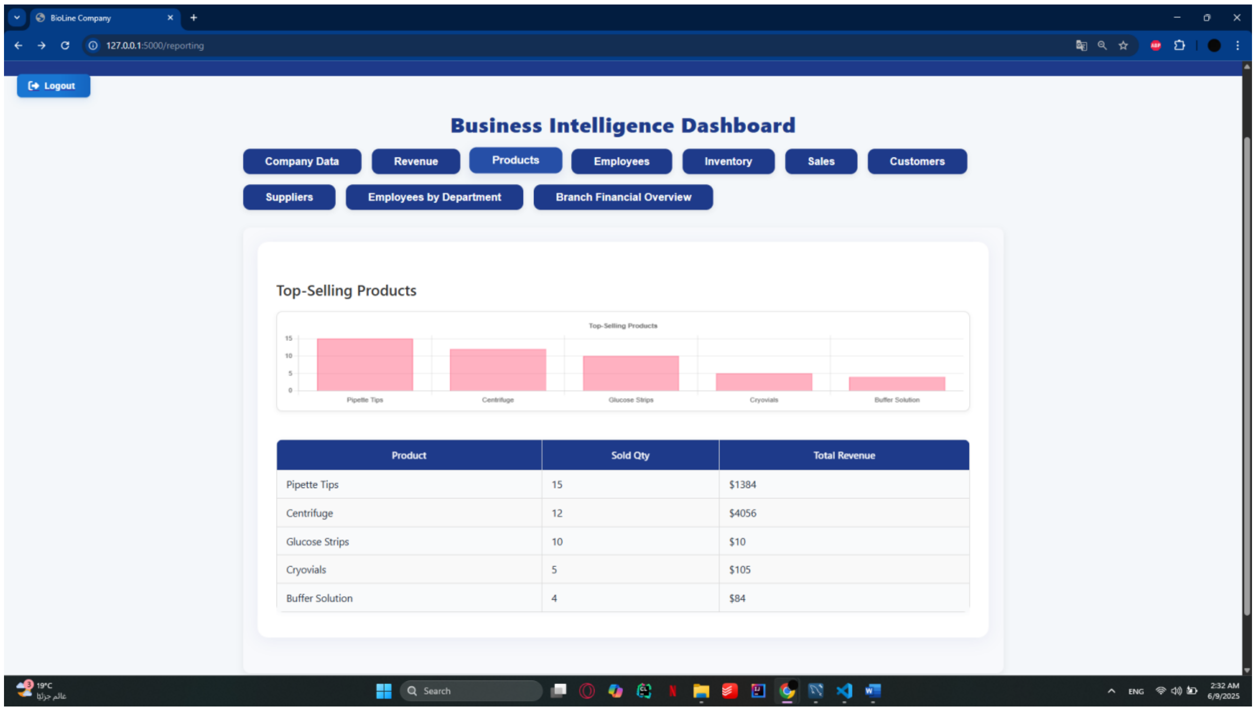
*Figure 24:Query1*



Revenue Report

| Branch | Total Revenue | Sales Count |
|---|---|---|
| BioLine Ramallah | $415 | 1 |
| BioLine Bethlehem | $5272 | 4 |
| BioLine Hebron | $271 | 1 |

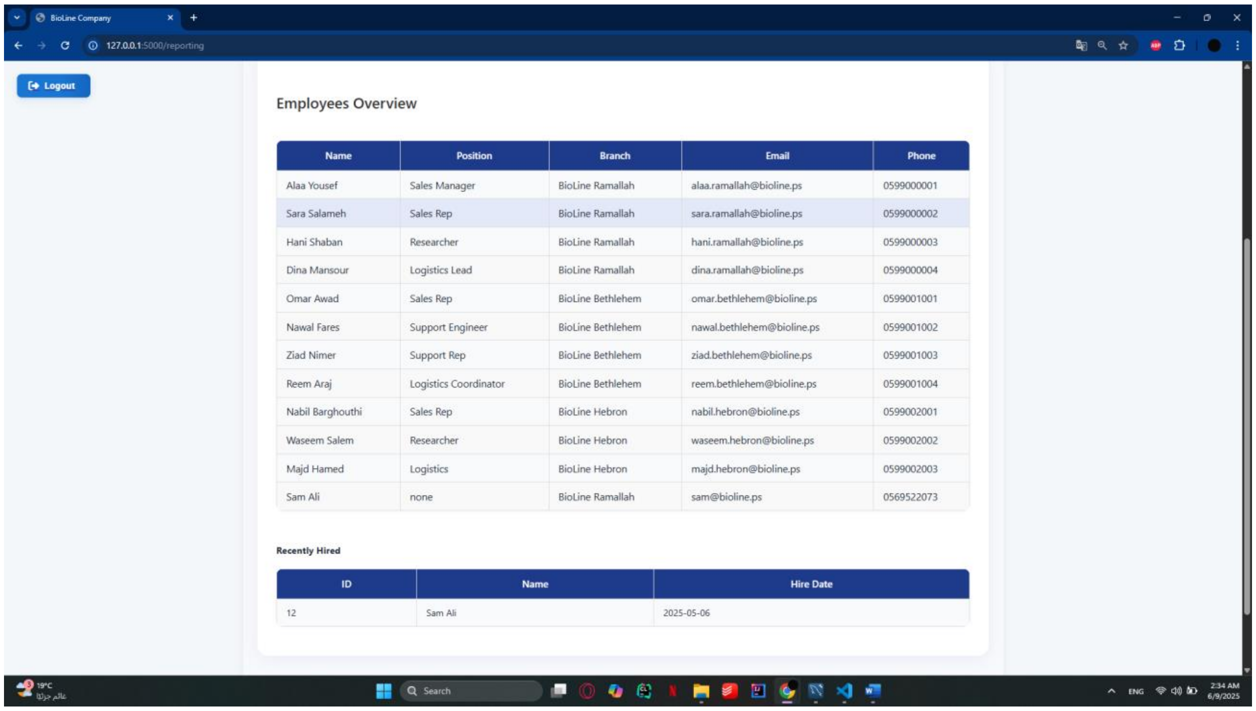
*Figure 25-Query 2*



Top-Selling Products

| Product | Sold Qty | Total Revenue |
|---|---|---|
| Pipette Tips | 15 | $1384 |
| Centrifuge | 12 | $4056 |
| Glucose Strips | 10 | $10 |
| Cryovials | 5 | $105 |
| Buffer Solution | 4 | $84 |

*Figure 26- Query 3*



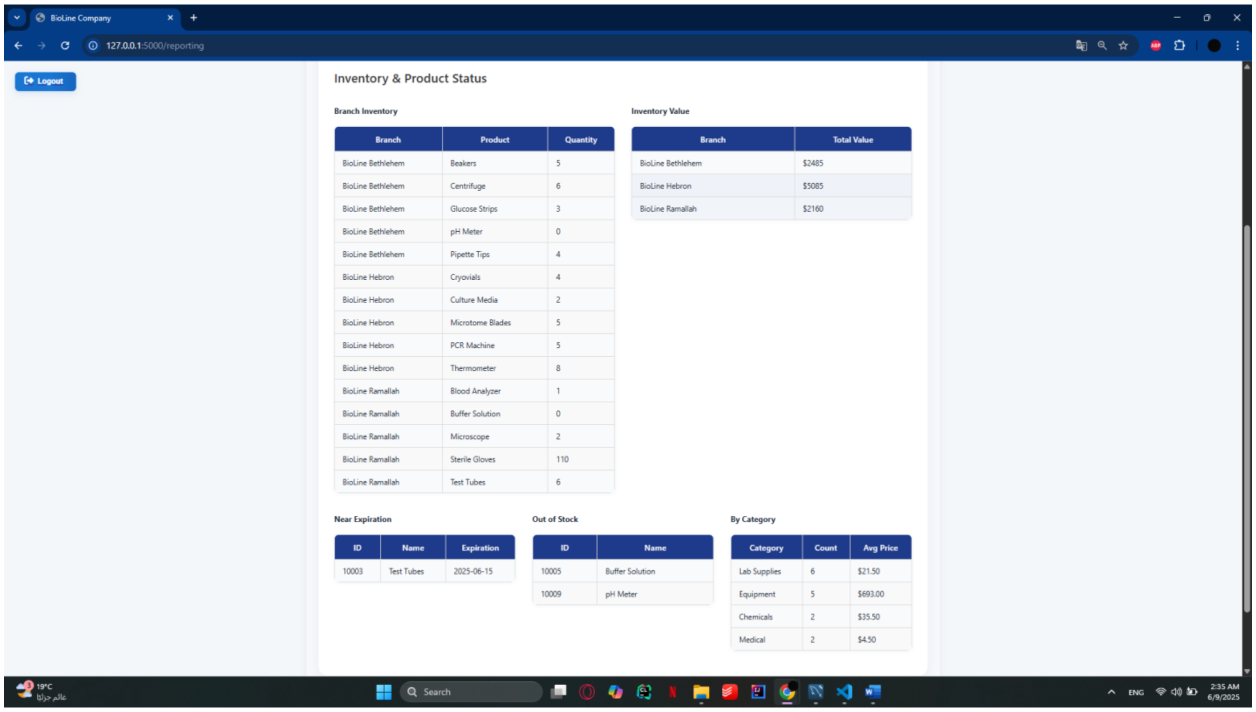
Employees Overview

| Name | Position | Branch | Email | Phone |
|---|---|---|---|---|
| Alaa Yousef | Sales Manager | BioLine Ramallah | alaa.ramallah@bioline.ps | 0599000001 |
| Sara Salameh | Sales Rep | BioLine Ramallah | sara.ramallah@bioline.ps | 0599000002 |
| Hani Shaban | Researcher | BioLine Ramallah | hani.ramallah@bioline.ps | 0599000003 |
| Dina Mansour | Logistics Lead | BioLine Ramallah | dina.ramallah@bioline.ps | 0599000004 |
| Omar Awad | Sales Rep | BioLine Bethlehem | omar.bethlehem@bioline.ps | 0599001001 |
| Nawal Fares | Support Engineer | BioLine Bethlehem | nawal.bethlehem@bioline.ps | 0599001002 |
| Ziad Nimer | Support Rep | BioLine Bethlehem | ziad.bethlehem@bioline.ps | 0599001003 |
| Reem Araj | Logistics Coordinator | BioLine Bethlehem | reem.bethlehem@bioline.ps | 0599001004 |
| Nabil Barghouthi | Sales Rep | BioLine Hebron | nabil.hebron@bioline.ps | 0599002001 |
| Waseem Salem | Researcher | BioLine Hebron | waseem.hebron@bioline.ps | 0599002002 |
| Majd Hamed | Logistics | BioLine Hebron | majd.hebron@bioline.ps | 0599002003 |
| Sam Ali | none | BioLine Ramallah | sam@bioline.ps | 0569522073 |

Recently Hired

| ID | Name | Hire Date |
|---|---|---|
| 12 | Sam Ali | 2025-05-06 |

*Figure 27 -Query 4*



Inventory & Product Status

Branch Inventory

| Branch | Product | Quantity |
|---|---|---|
| BioLine Bethlehem | Beakers | 5 |
| BioLine Bethlehem | Centrifuge | 6 |
| BioLine Bethlehem | Glucose Strips | 3 |
| BioLine Bethlehem | pH Meter | 0 |
| BioLine Bethlehem | Pipette Tips | 4 |
| BioLine Hebron | Cryovials | 4 |
| BioLine Hebron | Culture Media | 2 |
| BioLine Hebron | Microtome Blades | 5 |
| BioLine Hebron | PCR Machine | 5 |
| BioLine Hebron | Thermometer | 8 |
| BioLine Ramallah | Blood Analyzer | 1 |
| BioLine Ramallah | Buffer Solution | 0 |
| BioLine Ramallah | Microscope | 2 |
| BioLine Ramallah | Sterile Gloves | 110 |
| BioLine Ramallah | Test Tubes | 6 |

Inventory Value

| Branch | Total Value |
|---|---|
| BioLine Bethlehem | $2485 |
| BioLine Hebron | $5085 |
| BioLine Ramallah | $2160 |

Near Expiration

| ID | Name | Expiration |
|---|---|---|
| 10003 | Test Tubes | 2025-06-15 |

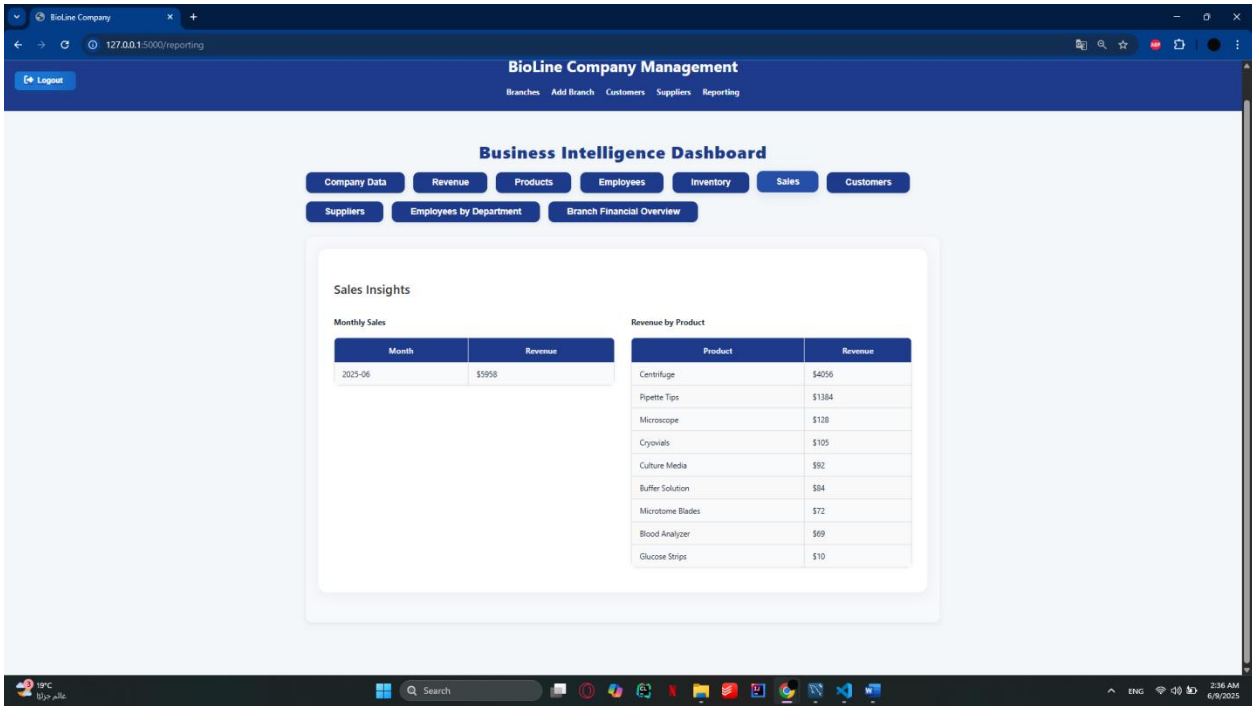
Out of Stock

| ID | Name |
|---|---|
| 10005 | Buffer Solution |
| 10009 | pH Meter |

By Category

| Category | Count | Avg Price |
|---|---|---|
| Lab Supplies | 6 | $21.50 |
| Equipment | 5 | $693.00 |
| Chemicals | 2 | $35.50 |
| Medical | 2 | $4.50 |

*Figure 28 Query 5*



Sales Insights

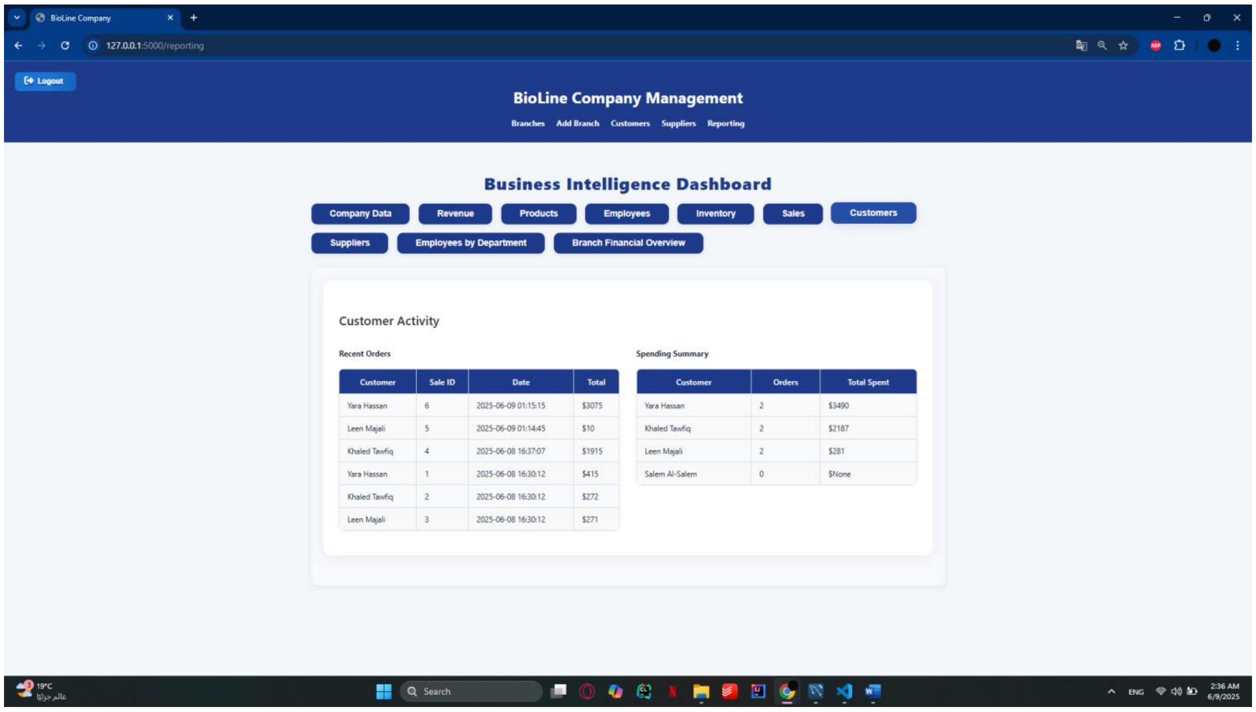
Monthly Sales

| Month | Revenue |
|---|---|
| 2025-06 | $5958 |

Revenue by Product

| Product | Revenue |
|---|---|
| Centrifuge | $4056 |
| Pipette Tips | $1384 |
| Microscope | $128 |
| Cryovials | $105 |
| Culture Media | $92 |
| Buffer Solution | $84 |
| Microtome Blades | $72 |
| Blood Analyzer | $69 |
| Glucose Strips | $10 |

*Figure 29 - Query 6*



Customer Activity

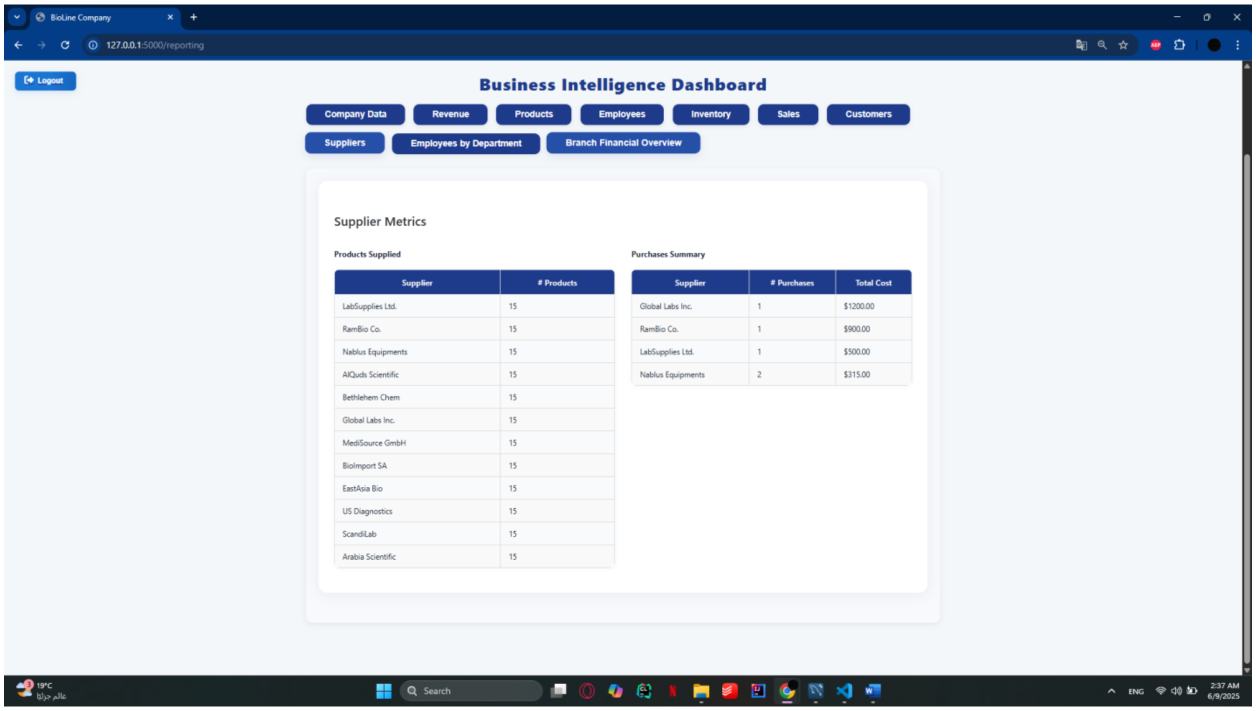
Recent Orders

| Customer | Sale ID | Date | Total |
|---|---|---|---|
| Yara Hassan | 6 | 2025-06-09 01:15:15 | $3075 |
| Leen Majali | 5 | 2025-06-09 01:14:45 | $10 |
| Khaled Tawfiq | 4 | 2025-06-08 16:37:07 | $1915 |
| Yara Hassan | 1 | 2025-06-08 16:30:12 | $415 |
| Khaled Tawfiq | 2 | 2025-06-08 16:30:12 | $272 |
| Leen Majali | 3 | 2025-06-08 16:30:12 | $271 |

Spending Summary

| Customer | Orders | Total Spent |
|---|---|---|
| Yara Hassan | 2 | $3490 |
| Khaled Tawfiq | 2 | $2187 |
| Leen Majali | 2 | $281 |
| Salem Al-Salem | 0 | $None |

*Figure 30 - Query 7*



Business Intelligence Dashboard

Supplier Metrics

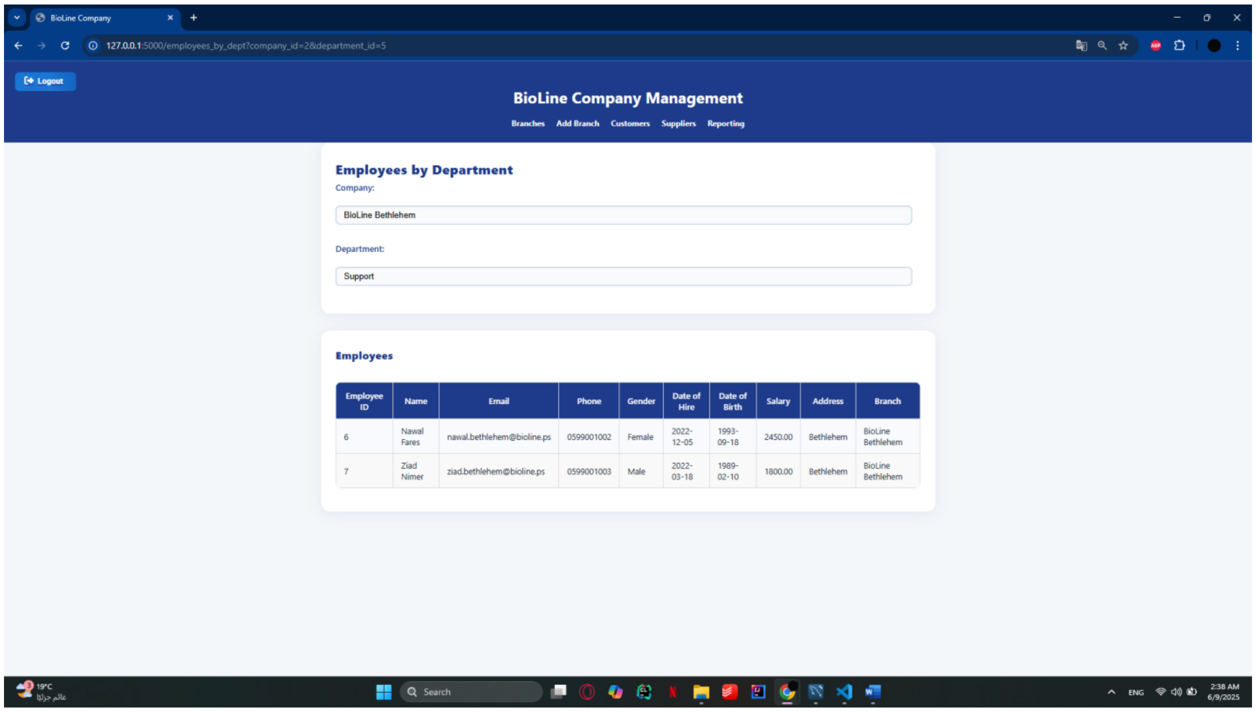
Products Supplied

| Supplier | # Products |
|---|---|
| LabSupplies Ltd. | 15 |
| RamBio Co. | 15 |
| Nablus Equipments | 15 |
| AlQuds Scientific | 15 |
| Bethlehem Chem | 15 |
| Global Labs Inc. | 15 |
| MediSource GmbH | 15 |
| BioImport SA | 15 |
| EastAsia Bio | 15 |
| US Diagnostics | 15 |
| ScandiLab | 15 |
| Arabia Scientific | 15 |

Purchases Summary

| Supplier | # Purchases | Total Cost |
|---|---|---|
| Global Labs Inc. | 1 | $1200.00 |
| RamBio Co. | 1 | $900.00 |
| LabSupplies Ltd. | 1 | $500.00 |
| Nablus Equipments | 2 | $315.00 |

*Figure 31 - Query 8*


BioLine Company
127.0.0.1:5000/employees_by_dept?company_id=2&department_id=5
Logout

BioLine Company Management

Employees by Department

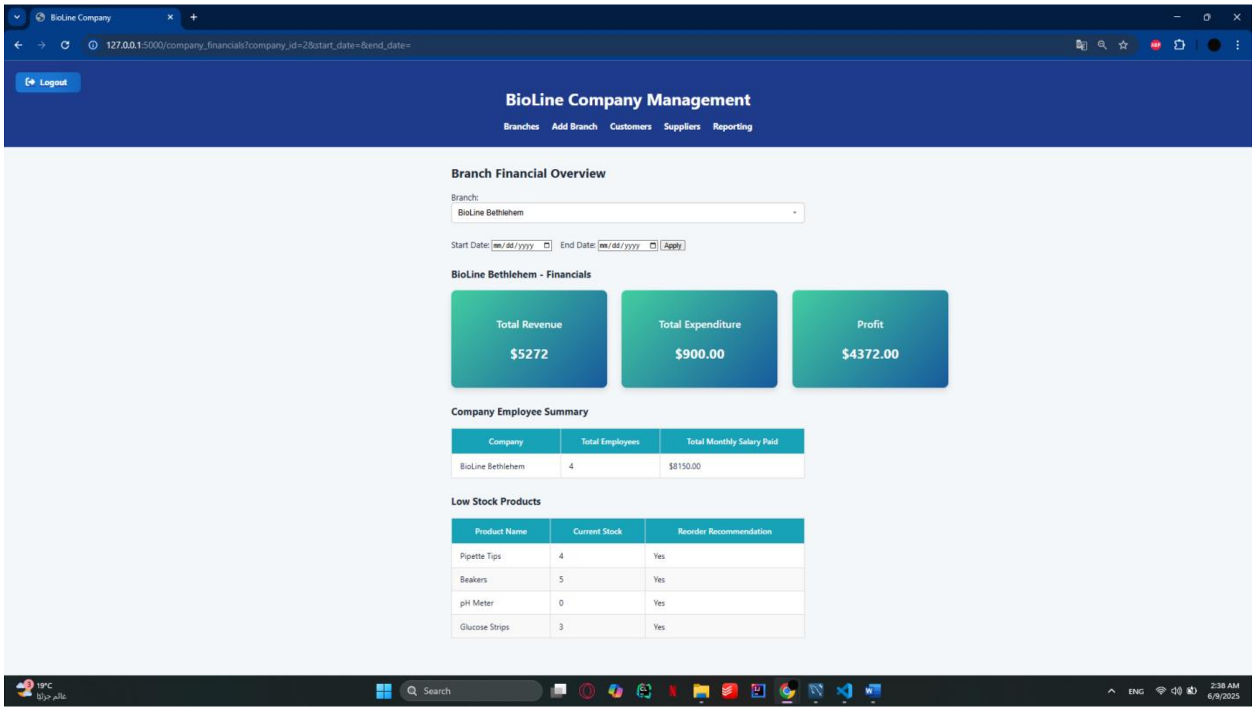
Company: BioLine Bethlehem

Department: Support

Employees

| Employee ID | Name | Email | Phone | Gender | Date of Hire | Date of Birth | Salary | Address | Branch |
|---|---|---|---|---|---|---|---|---|---|
| 6 | Nawal Fares | nawal.bethlehem@bioline.ps | 0599001002 | Female | 2022-12-05 | 1993-09-18 | 2450.00 | Bethlehem | BioLine Bethlehem |
| 7 | Ziad Nimer | ziad.bethlehem@bioline.ps | 0599001003 | Male | 2022-03-18 | 1989-02-10 | 1800.00 | Bethlehem | BioLine Bethlehem |

*Figure 32 - Query 9*


BioLine Company
127.0.0.1:5000/company_financials?company_id=2&start_date=&end_date=
Logout

BioLine Company Management

Branches Add Branch Customers Suppliers Reporting

Branch Financial Overview

Branch: BioLine Bethlehem

Start Date: mm/dd/yyyy End Date: mm/dd/yyyy Apply

BioLine Bethlehem - Financials

Total Revenue $5272

Total Expenditure $900.00

Profit $4372.00

Company Employee Summary

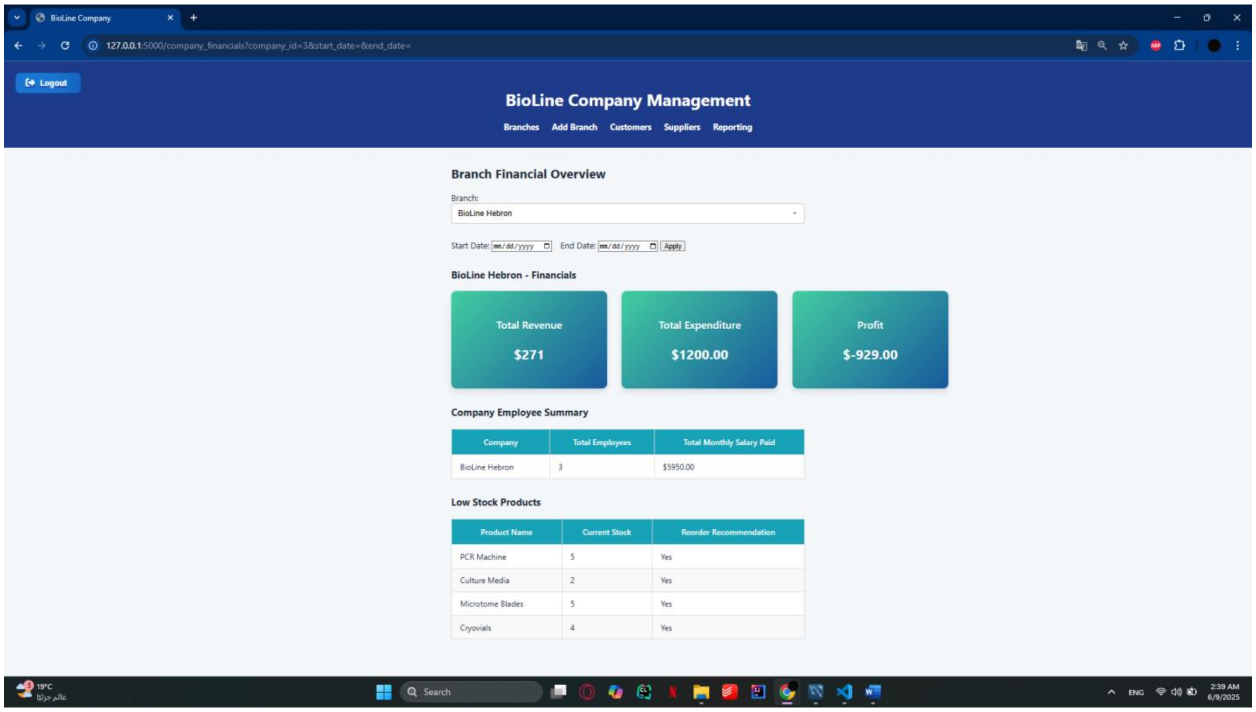
| Company | Total Employees | Total Monthly Salary Paid |
|---|---|---|
| BioLine Bethlehem | 4 | $8150.00 |

Low Stock Products

| Product Name | Current Stock | Reorder Recommendation |
|---|---|---|
| Pipette Tips | 4 | Yes |
| Beakers | 5 | Yes |
| pH Meter | 0 | Yes |
| Glucose Strips | 3 | Yes |

*Figure 33 - Financial Overview*

.


BioLine Company
127.0.0.1:5000/company_financials?company_id=3&start_date=&end_date=
Logout

BioLine Company Management

Branches Add Branch Customers Suppliers Reporting

Branch Financial Overview

Branch: BioLine Hebron

Start Date: mm/dd/yyyy End Date: mm/dd/yyyy Apply

BioLine Hebron - Financials

Total Revenue $271

Total Expenditure $1200.00

Profit $-929.00

Company Employee Summary

| Company | Total Employees | Total Monthly Salary Paid |
|---|---|---|
| BioLine Hebron | 3 | $5950.00 |

Low Stock Products

| Product Name | Current Stock | Reorder Recommendation |
|---|---|---|
| PCR Machine | 5 | Yes |
| Culture Media | 2 | Yes |
| Microtome Blades | 5 | Yes |
| Cryovials | 4 | Yes |

*Figure 34 - Financial Overview2*

## Conclusion

The BioLineCompany Management System project demonstrates a fully integrated solution for managing operations across company branches, departments, inventory, employees, customers, and suppliers. This system combines a normalized MySQL database with a clean, modular Flask backend and responsive web interface, ensuring data consistency, ease of use, and real-time reporting.

Key achievements include:

- Complete CRUD functionality across 10+ entities
- Real-time business intelligence dashboards
- Automatic inventory and employee count tracking via triggers
- Financial summaries and audit-friendly reporting

Through normalization (1NF, 2NF, 3NF), modular code structure, and relational integrity, the system is built to be reliable, extensible, and maintainable. Every aspect — from stock level updates to employee management and financial analytic — is tightly integrated for smooth operations.

This project lays a scalable foundation for future enhancements, such as:

- User authentication and access control
- Advanced forecasting using machine learning
- API integration with third-party tools (e.g., ERP, CRM)
- Mobile app version or progressive web app support

In conclusion, the system offers a professional-grade solution to manage and monitor the core operations of a growing biomedical supply company and serves as a model for similar enterprise management systems.