**Website Documentation: VendorBender - Vendor and Contract Management System**

**1. Project Overview**

The **VendorBender** website is a comprehensive system for managing vendors, contracts, budgets, and vendor performance. The platform allows users (vendors and admins) to register, log in, create and manage contracts, track budgets, evaluate vendor performance, and more.

**2. Features**

1. **Vendor Registration**:
   * Vendors can register by providing their details such as name, email, phone, service category, and compliance certification.
   * Passwords are securely hashed before being stored in the database.
2. **Vendor Login**:
   * Vendors can log in using their email and password.
   * If the credentials are valid, they are redirected to the vendor dashboard.
3. **Contract Management**:
   * Admins can create contracts for vendors, specifying contract terms, expiry date, and status.
   * Contracts can be viewed and managed.
4. **Budget Management**:
   * Admins can add and track budgets for various departments, with the system automatically calculating remaining amounts.
   * View a list of existing budgets and update them as needed.
5. **Vendor Performance Evaluation**:
   * Admins can rate vendor performance based on criteria like service quality, delivery, and pricing.
   * Vendors can see their performance reports on their dashboard.
6. **Dashboard**:
   * After logging in, vendors can view and manage their contracts, budget, and performance data from a central dashboard.
7. **Purchase Orders**:
   * Admins can add and track purchase orders against their contract IDs and Vendor IDs

**3. Technologies Used**

* **Frontend**:
  + HTML, CSS, JavaScript (ES6+)
  + Bootstrap 5 (for responsive design)
  + particles.js (for background particle effects)
* **Backend**:
  + Node.js with **Express.js**
  + **MySQL** for database management
  + **bcryptjs** for password hashing
  + **cors** for enabling Cross-Origin Resource Sharing (CORS)
* **Database**:
  + **MySQL** (Database name: VendorManagement)

**4. DataBase Structure:**

CREATE DATABASE VendorManagement;

USE VendorManagement;

-- Vendor

CREATE TABLE Vendor (

VendorID INT AUTO\_INCREMENT PRIMARY KEY,

Name VARCHAR(100) NOT NULL,

Email VARCHAR(100) NOT NULL,

Phone VARCHAR(15) NOT NULL,

ServiceCategory VARCHAR(50),

ComplianceCertification VARCHAR(150)

);

ALTER TABLE VendorContactInfo

CHANGE ContactInfo Phone VARCHAR(20) NOT NULL;

alter table Vendor

add Password varchar(20) NOT NULL;

alter table Vendor

modify Password varchar(1500) NOT NULL;

-- Contract Table

CREATE TABLE Contract (

ContractID INT AUTO\_INCREMENT PRIMARY KEY,

VendorID INT NOT NULL,

Terms TEXT,

ExpiryDate DATE NOT NULL,

Status ENUM('Active', 'Pending Renewal', 'Expired') DEFAULT 'Active',

FOREIGN KEY (VendorID) REFERENCES Vendor(VendorID)

);

-- PurchaseOrder Table

CREATE TABLE PurchaseOrder (

POID INT AUTO\_INCREMENT PRIMARY KEY,

ContractID INT NOT NULL,

VendorID INT NOT NULL,

ItemDescription VARCHAR(255) NOT NULL,

Quantity INT NOT NULL,

TotalCost FLOAT NOT NULL,

FOREIGN KEY (ContractID) REFERENCES Contract(ContractID),

FOREIGN KEY (VendorID) REFERENCES Vendor(VendorID)

);

ALTER TABLE PurchaseOrder

MODIFY COLUMN ContractID INT DEFAULT 0;

-- Budget Table

CREATE TABLE Budget (

BudgetID INT AUTO\_INCREMENT PRIMARY KEY,

Department VARCHAR(100) NOT NULL,

AllocatedAmount FLOAT NOT NULL,

SpentAmount FLOAT NOT NULL,

RemainingAmount FLOAT AS (AllocatedAmount - SpentAmount) STORED

);

-- Performance Table

CREATE TABLE Performance (

PerformanceID INT AUTO\_INCREMENT PRIMARY KEY,

VendorID INT NOT NULL,

Rating FLOAT CHECK (Rating BETWEEN 1 AND 5),

Feedback TEXT,

Date DATE NOT NULL,

FOREIGN KEY (VendorID) REFERENCES Vendor(VendorID)

);

DELIMITER //

CREATE PROCEDURE RegisterVendor(

IN VendorName VARCHAR(100),

IN ContactInfo VARCHAR(100),

IN ServiceCategory VARCHAR(50),

IN ComplianceCert VARCHAR(150)

)

BEGIN

INSERT INTO Vendor (Name, ContactInfo, ServiceCategory, ComplianceCertification)

VALUES (VendorName, ContactInfo, ServiceCategory, ComplianceCert);

END //

DELIMITER ;

DELIMITER //

CREATE TRIGGER NotifyContractRenewals

BEFORE INSERT ON Contract

FOR EACH ROW

BEGIN

IF NEW.ExpiryDate <= DATE\_ADD(CURDATE(), INTERVAL 30 DAY) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Contract is nearing expiration. Notify stakeholders!';

END IF;

END //

DELIMITER ;

DELIMITER //

CREATE TRIGGER CheckBudgetLimits

BEFORE INSERT ON PurchaseOrder

FOR EACH ROW

BEGIN

DECLARE BudgetLimit FLOAT;

SELECT AllocatedAmount - SpentAmount INTO BudgetLimit

FROM Budget WHERE Department = 'Procurement'; -- Adjust department dynamically

IF NEW.TotalCost > BudgetLimit THEN

SIGNAL SQLSTATE '45000'

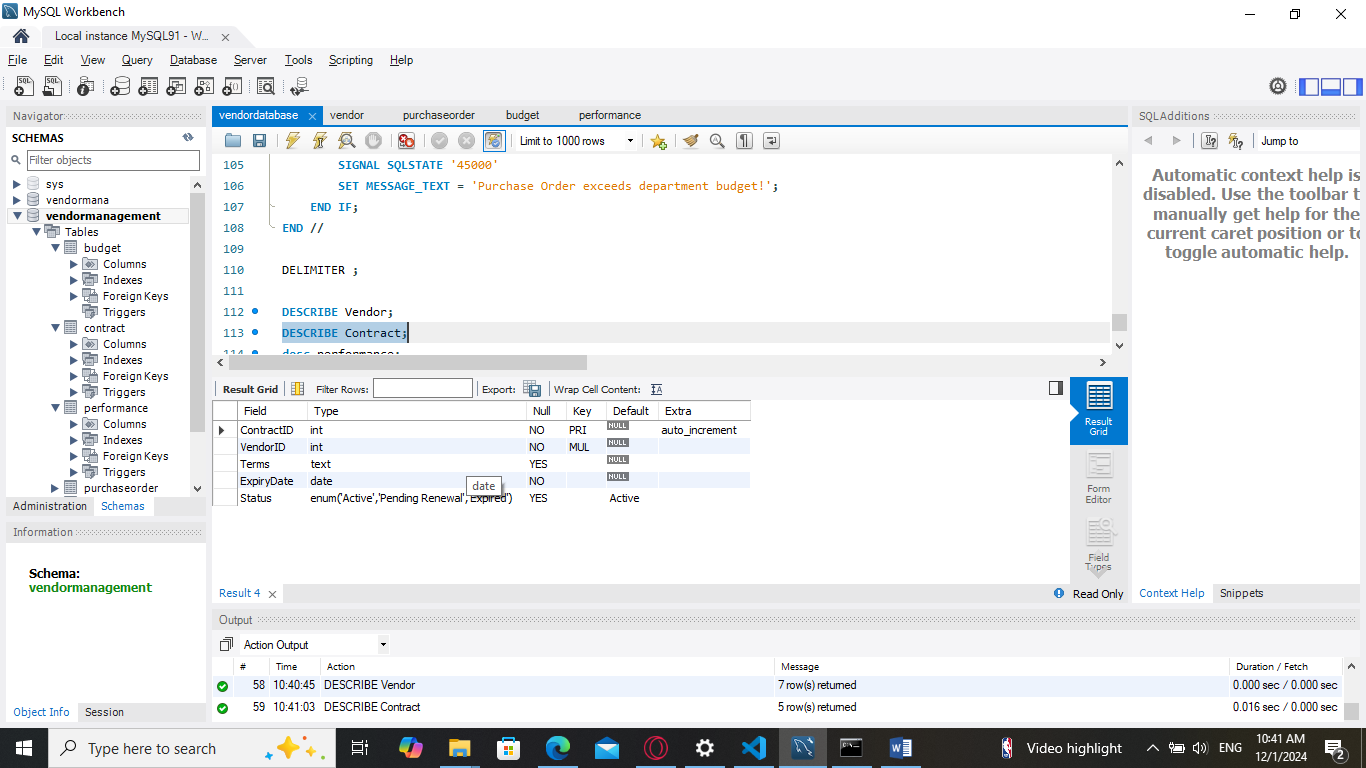
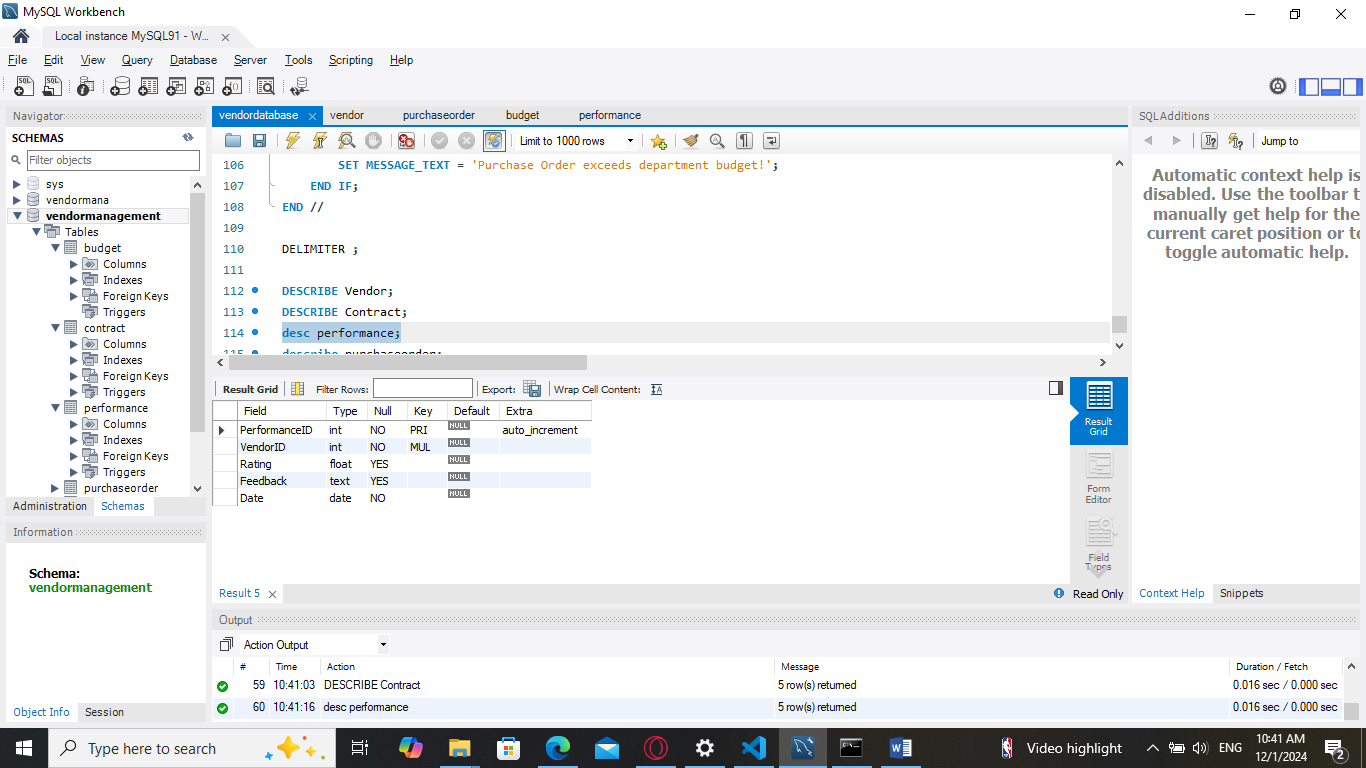
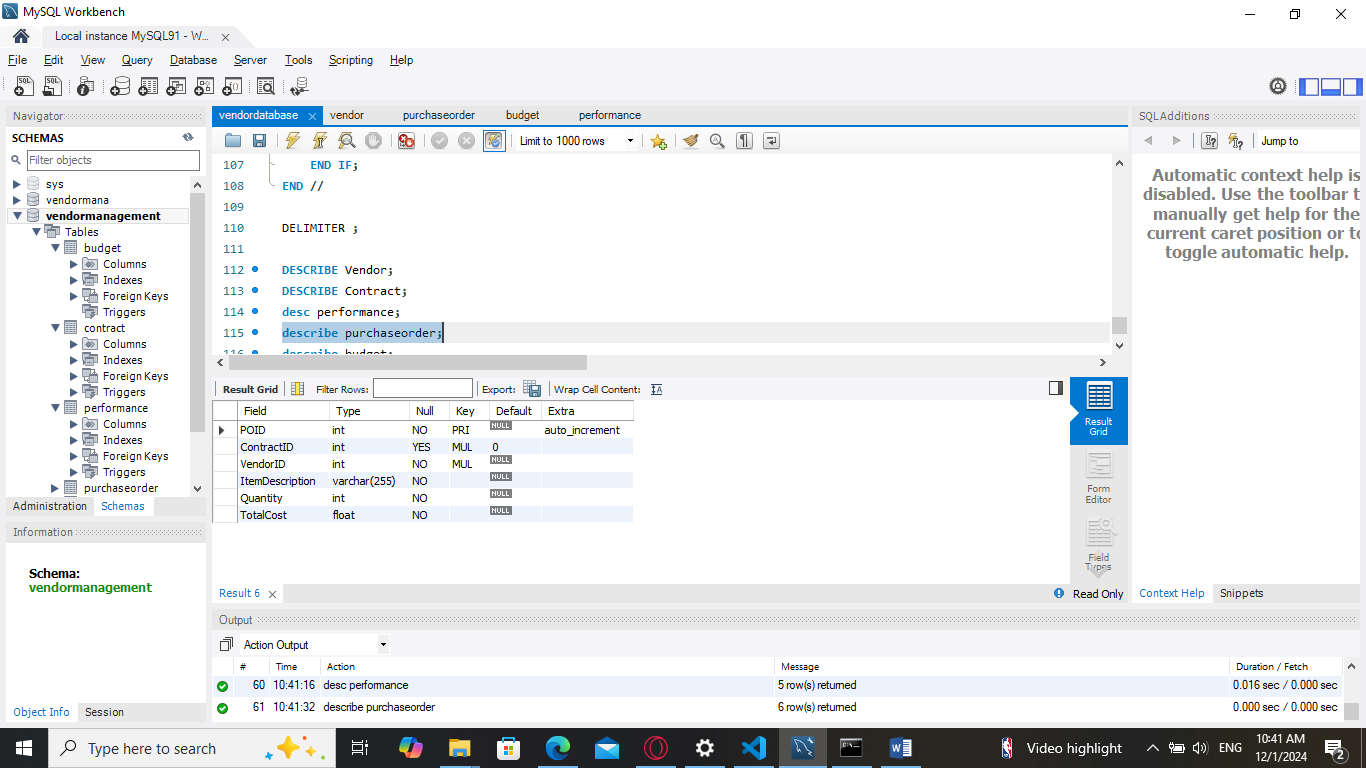
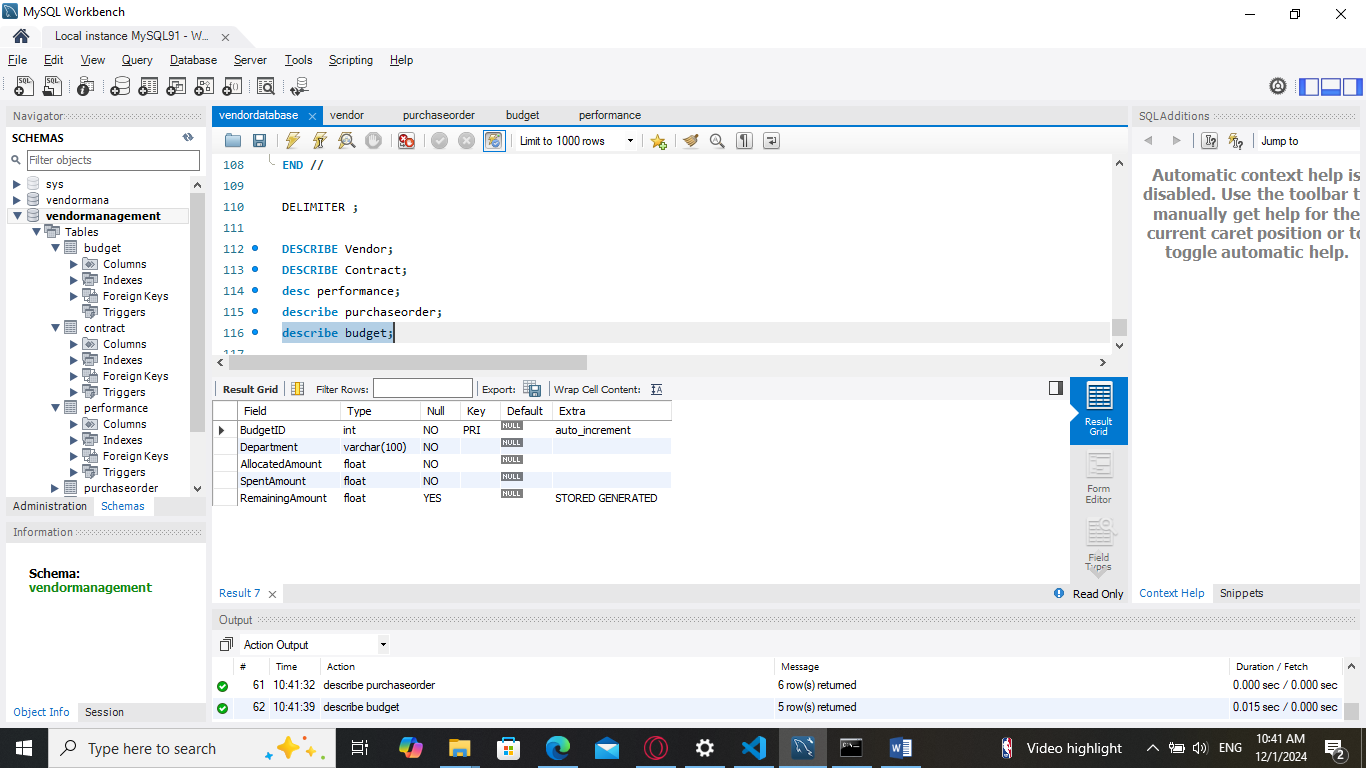
SET MESSAGE\_TEXT = 'Purchase Order exceeds department budget!';

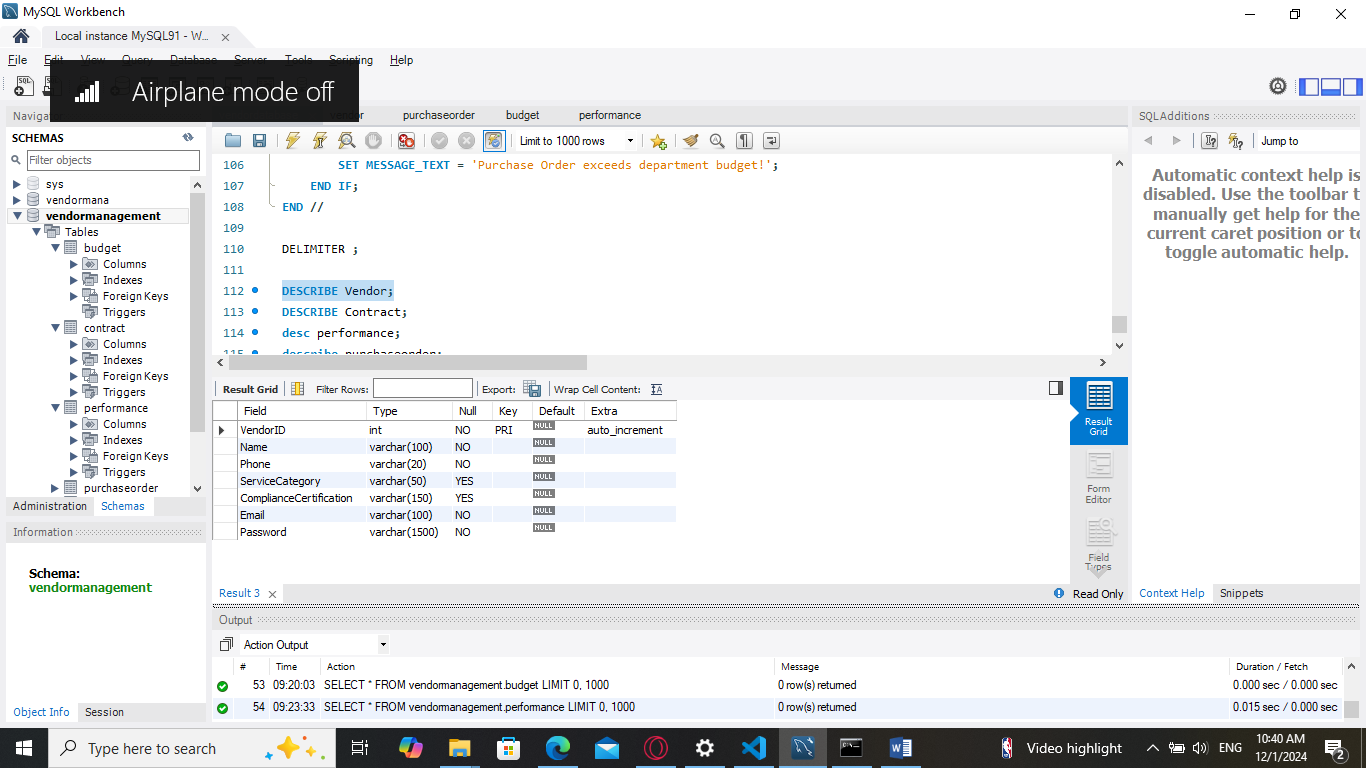
END IF;

END //

DELIMITER ;

**TABLE DESCRIPTIONS:**

****

****

Relational Schema:

* Vendor:

Vendor(VendorID (PK), Name, Email, Phone, ServiceCategory, ComplianceCertification, Password)

* Contract:

Contract(ContractID (PK), VendorID (FK), Terms, ExpiryDate, Status)

FK (VendorID) references Vendor(VendorID)

* PurchaseOrder:

PurchaseOrder(POID (PK), ContractID (FK), VendorID (FK), ItemDescription, Quantity, TotalCost)

FK (ContractID) references Contract(ContractID)

FK (VendorID) references Vendor(VendorID)

* Budget:

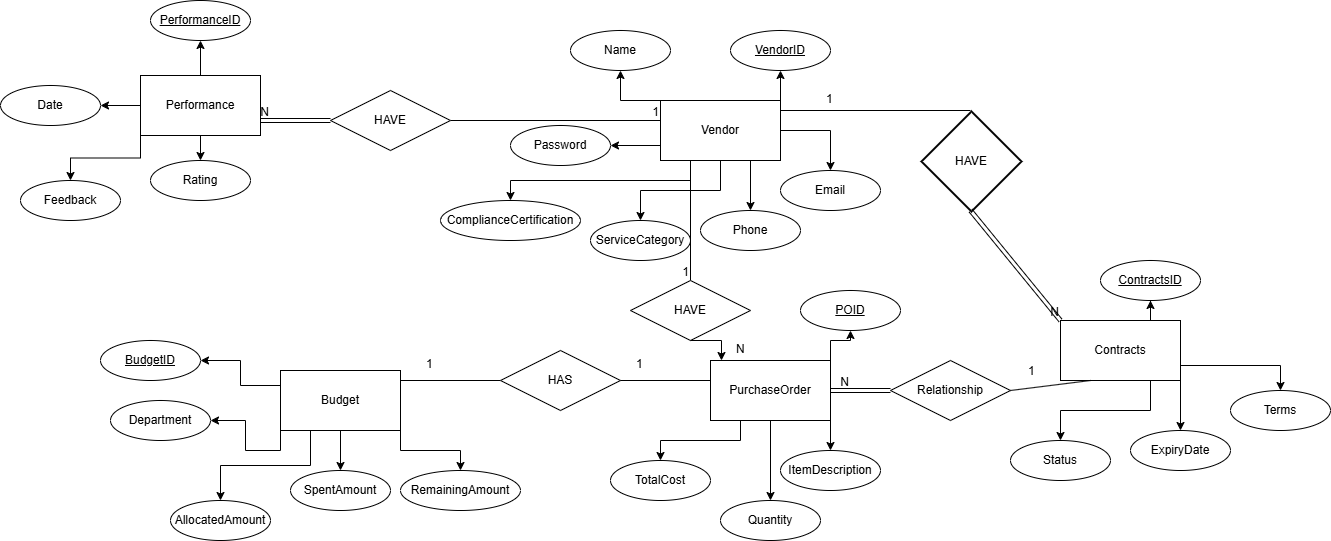
Budget(BudgetID (PK), Department, AllocatedAmount, SpentAmount, RemainingAmount (Computed))

* Performance:

Performance(PerformanceID (PK), VendorID (FK), Rating, Feedback, Date)

FK (VendorID) references Vendor(VendorID)

**ERD:**

****

FLOW:

[Landing Page]

|

v

[Register Page] <-> [Login Page]

|

v

[Dashboard Page]

|-------------------------|

| |

v v

[Contracts Page] [Budget Page] [Vendor Performance Page]

**CONCLUSION:**

**VendorBender** provides a comprehensive, easy-to-use platform for managing vendors, contracts, budgets, and vendor performance. It allows vendors to track their contracts, manage departmental budgets, and monitor performance evaluations. Admins can create, update, and manage vendors and associated data, ensuring seamless operation of the vendor relationship management process.

This system improves efficiency, reduces manual errors, and ensures better vendor relations through a centralized platform.