VENDOR AND CONTRACT MANAGEMENT SYSTEM FOR EVENTS

ABSTRACT :

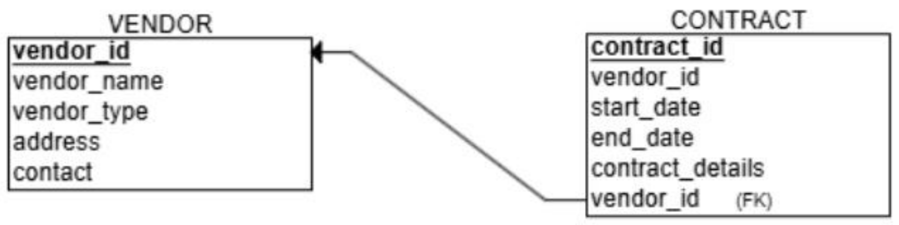
The Vendor and Contract Management System simplifies managing vendor data and associated contracts. It enables efficient registration, updates, and retrieval of vendor information, along with managing contract details tied to each vendor. The system uses a MySQL database to store vendor and contract information and provides a console-based interface for users to perform CRUD operations.

Key components include:

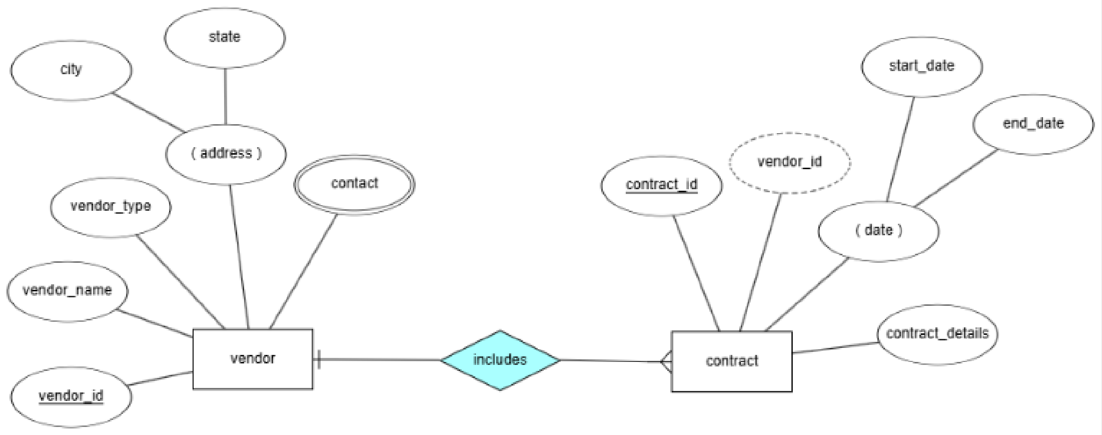
1. Vendor Table: Stores vendor ID, name, type, address, and contact.
2. Contract Table: Stores contract ID, vendor ID (foreign key), start/end dates, and details.

The system streamlines vendor registration, contract creation, and data updates, ensuring smooth and transparent vendor relationship management.

SCHEMA DIAGRAM :



ER DIAGRAM :



PROGRAM :

(com.org.controller) – VendorContractController.java

```
package com.org.controller;
import com.org.model.VendorContract;
import com.org.service.Service;
import com.org.service.ServiceImpl;
import java.sql.SQLException;
import java.util.Scanner;
public class VendorContractController {
private static Scanner sn = new Scanner(System.in);
public static void main(String[] args) {
VendorContract vendor = new VendorContract();
try {
Service sv = new ServiceImpl();
vendor.db_connect
boolean exe = true;
System.out.println("Connection successful..");
while (exe) {
System.out.println(" 1. Register Vendor \n 2. View Vendor Details \n 3. Update Vendor Info \n 4. Create
Contract \n 5. View Contract \n 6. Update Contract \n 7. View All Vendors \n 8. Exit");
int input = sn.nextInt();
switch (input) {
case 1:
sv.registerVendor();
System.out.println("Vendor Registered.");
break;
case 2:
sv.viewVendorDetails();
System.out.println("Vendor Details Viewed.");
break;
case 3:
sv.updateVendorInfo();
System.out.println("Vendor Info Updated.");
break;
case 4:
sv.createContract();
System.out.println("Contract Created.");
break;
case 5:
sv.viewContract();
System.out.println("Contract Details Viewed.");
break;
case 6:
sv.updateContract();
System.out.println("Contract Updated.");
break;
case 7:
sv.viewAllVendors();
System.out.println("All Vendor Details Viewed.");
break;
case 8:
exe = false;
System.out.println("Exiting the system...");
break;
default:
System.out.println("Invalid choice. Please enter a valid option.");
}}
} catch (SQLException e) {
e.printStackTrace();
}}}
```

(com.org.model) – VendorContract.java

```
package com.org.model;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class VendorContract {

public static Connection db_connect() throws SQLException {

return
DriverManager.getConnection("jdbc:mysql://localhost:3306/workusecase","root","W7301@jqir#");

}}
```

(com.org.service) – Service.java

```
package com.org.service;

public interface Service {

// Vendor Management Methods

void registerVendor();

void viewVendorDetails();

void updateVendorInfo();

void viewAllVendors();

// Contract Management Methods

void createContract();

void viewContract();

void updateContract();

}
```

(com.org.service) – ServiceImpl.java

```
package com.org.service;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;
import com.org.model.VendorContract;
public class ServiceImpl implements Service{
private static Scanner sn = new Scanner(System.in);
@Override
public void registerVendor() {
try {
Connection conn = VendorContract.db_connect();
System.out.println("Enter vendor id:");
int vendorId = sn.nextInt();
System.out.println("Enter vendor name:");
String name = sn.next();
System.out.println("Enter vendor type (1. Supplier 2. Service Provider):");
int typeId = sn.nextInt();
String type = (typeId == 1) ? "Supplier" : "Service Provider";
System.out.println("Enter vendor address:");
String address = sn.next();
System.out.println("Enter vendor contact number:");
String contact = sn.next();
String sql = "INSERT INTO vendor(vendor_id, vendor_name, vendor_type, address, contact) VALUES
(?,?,?,?,?)";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setInt(1, vendorId);
pstmt.setString(2, name);
pstmt.setString(3, type);
pstmt.setString(4, address);
pstmt.setString(5, contact);
pstmt.executeUpdate();
System.out.println("Vendor registered successfully.");
} catch (SQLException e) {
e.printStackTrace();
}}
@Override
public void viewVendorDetails() {
try {
System.out.println("Enter vendor id:");
int vendorId = sn.nextInt();
Connection conn = VendorContract.db_connect();
String sql = "SELECT * FROM vendor WHERE vendor_id = ?";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setInt(1, vendorId);
ResultSet rs = pstmt.executeQuery();
if (rs.next()) {
System.out.println("Vendor ID: " + rs.getInt("vendor_id"));
System.out.println("Vendor Name: " + rs.getString("vendor_name"));
System.out.println("Vendor Type: " + rs.getString("vendor_type"));
System.out.println("Address: " + rs.getString("address"));
System.out.println("Contact: " + rs.getString("contact"));
} else {
System.out.println("Vendor not found.");
}
} catch (SQLException e) {
e.printStackTrace();
}}
@Override
public void updateVendorInfo() {
try {
System.out.println("Enter vendor id:");
```

```
int vendorId = sn.nextInt();
System.out.println("Enter new address:");
String address = sn.next();
System.out.println("Enter new contact number:");
String contact = sn.next();
Connection conn = VendorContract.db_connect();
String sql = "UPDATE vendor SET address = ?, contact = ? WHERE vendor_id = ?";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setString(1, address);
pstmt.setString(2, contact);
pstmt.setInt(3, vendorId);
pstmt.executeUpdate();
System.out.println("Vendor information updated successfully.");
} catch (SQLException e) {
e.printStackTrace();
}}
@Override
public void createContract() {
try {
System.out.println("Enter contract id:");
int contractId = sn.nextInt();
System.out.println("Enter vendor id:");
int vendorId = sn.nextInt();
System.out.println("Enter contract details:");
String details = sn.next();
System.out.println("Enter contract start date (YYYY-MM-DD):");
String startDate = sn.next();
System.out.println("Enter contract end date (YYYY-MM-DD):");
String endDate = sn.next();
Connection conn = VendorContract.db_connect();
String sql = "INSERT INTO contract(contract_id, vendor_id, contract_details, start_date, end_date)
VALUES (?,?,?,?,?)";
PreparedStatement pstmt = conn.prepareStatement(sql);
```

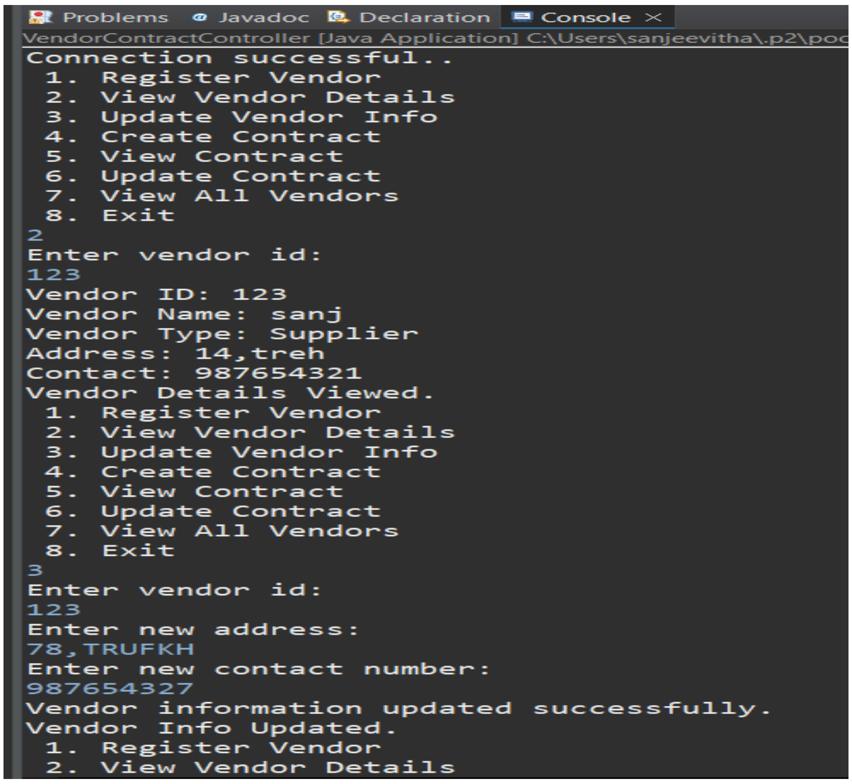
```
pstmt.setInt(1, contractId);
pstmt.setInt(2, vendorId);
pstmt.setString(3, details);
pstmt.setString(4, startDate);
pstmt.setString(5, endDate);
pstmt.executeUpdate();
System.out.println("Contract created successfully.");
} catch (SQLException e) {
e.printStackTrace();
}}
@Override
public void viewContract() {
try {
System.out.println("Enter contract id:");
int contractId = sn.nextInt();
Connection conn = VendorContract.db_connect();
String sql = "SELECT * FROM contract WHERE contract_id = ?";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setInt(1, contractId);
ResultSet rs = pstmt.executeQuery();
if (rs.next()) {
System.out.println("Contract ID: " + rs.getInt("contract_id"));
System.out.println("Vendor ID: " + rs.getInt("vendor_id"));
System.out.println("Contract Details: " + rs.getString("contract_details"));
System.out.println("Start Date: " + rs.getString("start_date"));
System.out.println("End Date: " + rs.getString("end_date"));
} else {
System.out.println("Contract not found.");
}
} catch (SQLException e) {
e.printStackTrace();
}}
```

```
@Override
public void updateContract() {
try {
System.out.println("Enter contract id:");
int contractId = sn.nextInt();
sn.nextLine();
System.out.println("Enter new contract details:");
String details = sn.next();
System.out.println("Enter new contract start date (YYYY-MM-DD):");
String startDate = sn.next();
System.out.println("Enter new contract end date (YYYY-MM-DD):");
String endDate = sn.next();
Connection conn = VendorContract.db_connect();
String sql = "UPDATE contract SET contract_details = ?, start_date = ?, end_date = ? WHERE
contract_id = ?";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setString(1, details);
pstmt.setString(2, startDate);
pstmt.setString(3, endDate);
pstmt.setInt(4, contractId);
pstmt.executeUpdate();
System.out.println("Contract updated successfully.");
} catch (SQLException e) {
e.printStackTrace();
}}
@Override
public void viewAllVendors() {
try {
Connection conn = VendorContract.db_connect();
PreparedStatement pstmt = conn.prepareStatement("SELECT * FROM vendor",
ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
ResultSet rs = pstmt.executeQuery();
if (!rs.next()) {
System.out.println("No vendor records found.");
} else {
rs.beforeFirst();
System.out.println("Vendor ID | Vendor Name | Vendor Type | Vendor Address | Vendor Contact Number");
while (rs.next()) {
int vendorId = rs.getInt("vendor_id");
String vendorName = rs.getString("vendor_name");
String vendorType = rs.getString("vendor_type");
String vendorAddress = rs.getString("address");
String vendorContact = rs.getString("contact");
System.out.println(vendorId + " | " + vendorName + " | " + vendorType + " | " + vendorAddress + " | " + vendorContact);
}}
} catch (SQLException e) {
e.printStackTrace();
}}
```

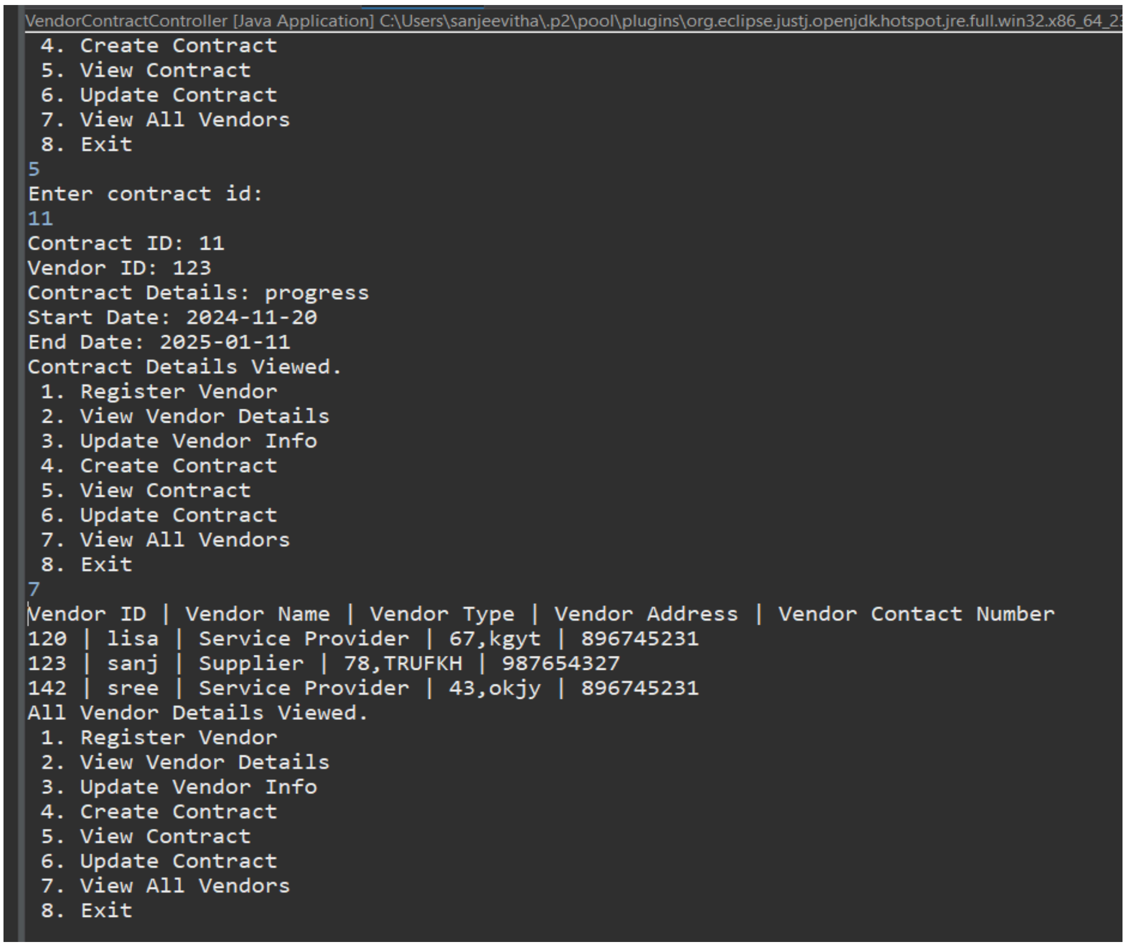
(TABLES CREATED)

```
create database workusecase;
use workusecase;

CREATE TABLE vendor ( vendor_id INT PRIMARY KEY, vendor_name VARCHAR(255),
 vendor_type VARCHAR(255), address VARCHAR(255),contact VARCHAR(15));
select * from vendor;

CREATE TABLE contract ( contract_id INT PRIMARY KEY AUTO_INCREMENT,vendor_id INT,
start_date DATE,end_date DATE,contract_details TEXT,  FOREIGN KEY (vendor_id)
REFERENCES vendor(vendor_id));
select * from contract;
```

OUTPUT :



```
VendorContractController [Java Application] C:\Users\sanjeevitha\.p2\poo
Connection successful..
 1. Register Vendor
 2. View Vendor Details
 3. Update Vendor Info
 4. Create Contract
 5. View Contract
 6. Update Contract
 7. View All Vendors
 8. Exit
2
Enter vendor id:
123
Vendor ID: 123
Vendor Name: sanj
Vendor Type: Supplier
Address: 14,treh
Contact: 987654321
Vendor Details Viewed.
 1. Register Vendor
 2. View Vendor Details
 3. Update Vendor Info
 4. Create Contract
 5. View Contract
 6. Update Contract
 7. View All Vendors
 8. Exit
3
Enter vendor id:
123
Enter new address:
78,TRUFKH
Enter new contact number:
987654327
Vendor information updated successfully.
Vendor Info Updated.
 1. Register Vendor
 2. View Vendor Details
```



```
VendorContractController [Java Application] C:\Users\sanjeevitha\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_2
 4. Create Contract
 5. View Contract
 6. Update Contract
 7. View All Vendors
 8. Exit
5
Enter contract id:
11
Contract ID: 11
Vendor ID: 123
Contract Details: progress
Start Date: 2024-11-20
End Date: 2025-01-11
Contract Details Viewed.
 1. Register Vendor
 2. View Vendor Details
 3. Update Vendor Info
 4. Create Contract
 5. View Contract
 6. Update Contract
 7. View All Vendors
 8. Exit
7
Vendor ID | Vendor Name | Vendor Type | Vendor Address | Vendor Contact Number
120 | lisa | Service Provider | 67,kgyt | 896745231
123 | sanj | Supplier | 78,TRUFKH | 987654327
142 | sree | Service Provider | 43,okjy | 896745231
All Vendor Details Viewed.
 1. Register Vendor
 2. View Vendor Details
 3. Update Vendor Info
 4. Create Contract
 5. View Contract
 6. Update Contract
 7. View All Vendors
 8. Exit
```