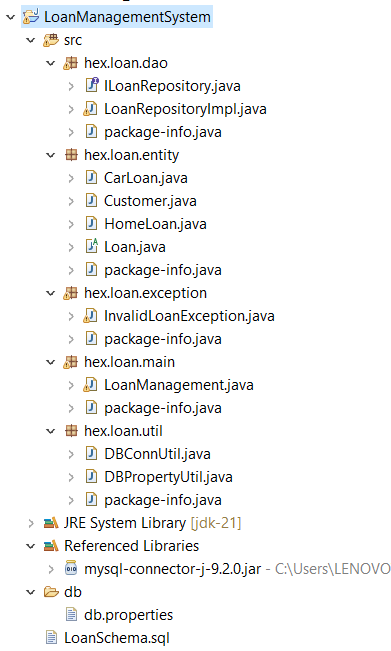
**Coding Challenge - Loan Management System**

**Github Submission Link:** <https://github.com/Shraddha-Gavkare/loan-management-system.git>  
**1. Introduction**  
  
The Loan Management System is a Java-based mini project designed to simulate real-world loan processing in a simplified, menu-driven console application. It allows users to apply for loans, calculate interest and EMI, check loan status, and manage repayments. The project follows an object-oriented approach and integrates with a MySQL database using JDBC and the DAO design pattern. This system demonstrates core Java concepts such as inheritance, encapsulation, abstraction, exception handling, and collections, making it a practical and educational tool for understanding backend development in enterprise-grade applications.

**2. Project Structure:**  
  
  
  
**3. Database Schema:**-- Create schema

CREATE DATABASE IF NOT EXISTS LoanDB;

USE LoanDB;

-- Create Customer table

CREATE TABLE IF NOT EXISTS Customer (

customer\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15) NOT NULL,

address VARCHAR(255),

credit\_score INT CHECK (credit\_score BETWEEN 300 AND 900)

);

-- Create Loan table

CREATE TABLE IF NOT EXISTS Loan (

loan\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT NOT NULL,

principal\_amount DOUBLE NOT NULL CHECK (principal\_amount > 0),

interest\_rate DOUBLE NOT NULL CHECK (interest\_rate > 0),

loan\_term INT NOT NULL CHECK (loan\_term > 0),

loan\_type ENUM('HomeLoan', 'CarLoan') NOT NULL,

loan\_status ENUM('Pending', 'Approved') DEFAULT 'Pending',

FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

select \* from Customer;

select \* from Loan;

USE LoanDB;

-- Insert Customer Data

INSERT INTO Customer (name, email, phone, address, credit\_score)

VALUES

('Aman Kumar', 'aman.kumar@example.com', '9876543210', 'Delhi, India', 720),

('Shravani Gavkare', 'shrau@gmail.com', '8766828200', 'Pune', 680),

('Rohan Sharma', 'rohan.sharma@gmail.com', '9001234567', 'Mumbai', 640);

-- Insert Loan Data

INSERT INTO Loan (customer\_id, principal\_amount, interest\_rate, loan\_term, loan\_type, loan\_status)

VALUES

(1, 500000, 9.2, 10, 'HomeLoan', 'Pending'),

(2, 300000, 10.5, 24, 'CarLoan', 'Pending'),

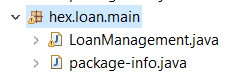
(3, 200000, 11.0, 12, 'CarLoan', 'Pending');

SELECT \* FROM Customer;

-- View loans with customer details

SELECT l.loan\_id, c.name, l.principal\_amount, l.interest\_rate, l.loan\_term, l.loan\_type, l.loan\_status

FROM Loan l

JOIN Customer c ON l.customer\_id = c.customer\_id;  
  
**4. Java Code (Package-wise):  
  
1) hex.loan.main**  
  
**LoanManagement.java**

package hex.loan.main;

import hex.loan.dao.ILoanRepository;

import hex.loan.dao.LoanRepositoryImpl;

import hex.loan.entity.\*;

import hex.loan.exception.InvalidLoanException;

import java.util.\*;

public class LoanManagement {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

ILoanRepository repo = new LoanRepositoryImpl();

while (true) {

System.out.println("\n=== \*Loan Management System\* ===");

System.out.println("1. Apply Loan:");

System.out.println("2. View All Loans:");

System.out.println("3. Get Loan By ID:");

System.out.println("4. Calculate Interest:");

System.out.println("5. Check Loan Status:");

System.out.println("6. Calculate EMI:");

System.out.println("7. Repay Loan:");

System.out.println("8. Exit\_\_\_");

System.out.print("Enter your choice: ");

int choice = sc.nextInt();

sc.nextLine(); // Consume newline

try {

switch (choice) {

case 1:

// Apply Loan

Customer cust = new Customer();

System.out.print("Enter Name: ");

cust.setName(sc.nextLine());

System.out.print("Enter Email: ");

cust.setEmail(sc.nextLine());

System.out.print("Enter Phone: ");

cust.setPhone(sc.nextLine());

System.out.print("Enter Address: ");

cust.setAddress(sc.nextLine());

System.out.print("Enter Credit Score: ");

cust.setCreditScore(sc.nextInt());

sc.nextLine();

System.out.print("Enter Principal Amount: ");

double principal = sc.nextDouble();

System.out.print("Enter Interest Rate: ");

double rate = sc.nextDouble();

System.out.print("Enter Loan Term (months): ");

int term = sc.nextInt();

sc.nextLine();

System.out.print("Choose Loan Type (HomeLoan/CarLoan): ");

String type = sc.nextLine();

Loan loan = new Loan(0, cust, principal, rate, term, type, "Pending") {};

repo.applyLoan(loan);

break;

case 2:

List<Loan> loans = repo.getAllLoan();

for (Loan l : loans) {

System.out.println(l);

}

break;

case 3:

System.out.print("Enter Loan ID: ");

int id = sc.nextInt();

Loan l = repo.getLoanById(id);

System.out.println(l);

break;

case 4:

System.out.print("Enter Loan ID: ");

int lid = sc.nextInt();

double interest = repo.calculateInterest(lid);

System.out.println("Calculated Interest: " + interest);

break;

case 5:

System.out.print("Enter Loan ID: ");

int sid = sc.nextInt();

String status = repo.loanStatus(sid);

System.out.println(status);

break;

case 6:

System.out.print("Enter Loan ID: ");

int eid = sc.nextInt();

double emi = repo.calculateEMI(eid);

System.out.printf("Monthly EMI: %.2f\n", emi);

break;

case 7:

System.out.print("Enter Loan ID: ");

int rid = sc.nextInt();

System.out.print("Enter Repayment Amount: ");

double amount = sc.nextDouble();

String result = repo.loanRepayment(rid, amount);

System.out.println(result);

break;

case 8:

System.out.println("Exiting... Goodbye!");

System.exit(0);

default:

System.out.println("Invalid choice.");

}

} catch (InvalidLoanException e) {

System.out.println("Error: " + e.getMessage());

} catch (Exception e) {

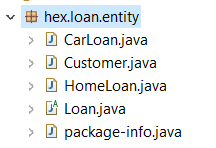
System.out.println("Unexpected error: " + e.getMessage());

}

}

}

}

**2.hex.loan.entity**  
  
**Customer.java**  
package hex.loan.entity;

public class Customer {

private int customerId;

private String name;

private String email;

private String phone;

private String address;

private int creditScore;

public Customer() {

}

public Customer(int customerId, String name, String email, String phone, String address, int creditScore) {

this.customerId = customerId;

this.name = name;

this.email = email;

this.phone = phone;

this.address = address;

this.creditScore = creditScore;

}

public int getCustomerId() { return customerId; }

public void setCustomerId(int customerId) { this.customerId = customerId; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public String getEmail() { return email; }

public void setEmail(String email) { this.email = email; }

public String getPhone() { return phone; }

public void setPhone(String phone) { this.phone = phone; }

public String getAddress() { return address; }

public void setAddress(String address) { this.address = address; }

public int getCreditScore() { return creditScore; }

public void setCreditScore(int creditScore) { this.creditScore = creditScore; }

@Override

public String toString() {

return "Customer [ID=" + customerId + ", Name=" + name + ", Email=" + email +

", Phone=" + phone + ", Address=" + address + ", CreditScore=" + creditScore + "]";

}

}

**Loan.java  
  
package** hex.loan.entity;

**public** **abstract** **class** Loan {

**private** **int** loanId;

**private** Customer customer;

**private** **double** principalAmount;

**private** **double** interestRate;

**private** **int** loanTerm;

**private** String loanType;

**private** String loanStatus;

**public** Loan() {

}

**public** Loan(**int** loanId, Customer customer, **double** principalAmount, **double** interestRate,

**int** loanTerm, String loanType, String loanStatus) {

**this**.loanId = loanId;

**this**.customer = customer;

**this**.principalAmount = principalAmount;

**this**.interestRate = interestRate;

**this**.loanTerm = loanTerm;

**this**.loanType = loanType;

**this**.loanStatus = loanStatus;

}

**public** **int** getLoanId() { **return** loanId; }

**public** **void** setLoanId(**int** loanId) { **this**.loanId = loanId; }

**public** Customer getCustomer() { **return** customer; }

**public** **void** setCustomer(Customer customer) { **this**.customer = customer; }

**public** **double** getPrincipalAmount() { **return** principalAmount; }

**public** **void** setPrincipalAmount(**double** principalAmount) { **this**.principalAmount = principalAmount; }

**public** **double** getInterestRate() { **return** interestRate; }

**public** **void** setInterestRate(**double** interestRate) { **this**.interestRate = interestRate; }

**public** **int** getLoanTerm() { **return** loanTerm; }

**public** **void** setLoanTerm(**int** loanTerm) { **this**.loanTerm = loanTerm; }

**public** String getLoanType() { **return** loanType; }

**public** **void** setLoanType(String loanType) { **this**.loanType = loanType; }

**public** String getLoanStatus() { **return** loanStatus; }

**public** **void** setLoanStatus(String loanStatus) { **this**.loanStatus = loanStatus; }

@Override

**public** String toString() {

**return** "Loan [LoanID=" + loanId + ", Customer=" + customer + ", PrincipalAmount=" + principalAmount +

", InterestRate=" + interestRate + ", LoanTerm=" + loanTerm + ", LoanType=" + loanType +

", LoanStatus=" + loanStatus + "]";

}

}

**CarLoan.java  
  
package** hex.loan.entity;

**public** **class** CarLoan **extends** Loan {

**private** String carModel;

**private** **int** carValue;

**public** CarLoan() {

**super**();

}

**public** CarLoan(**int** loanId, Customer customer, **double** principalAmount, **double** interestRate,

**int** loanTerm, String loanType, String loanStatus, String carModel, **int** carValue) {

**super**(loanId, customer, principalAmount, interestRate, loanTerm, loanType, loanStatus);

**this**.carModel = carModel;

**this**.carValue = carValue;

}

**public** String getCarModel() {

**return** carModel;

}

**public** **void** setCarModel(String carModel) {

**this**.carModel = carModel;

}

**public** **int** getCarValue() {

**return** carValue;

}

**public** **void** setCarValue(**int** carValue) {

**this**.carValue = carValue;

}

@Override

**public** String toString() {

**return** **super**.toString() + ", CarLoan [CarModel=" + carModel + ", CarValue=" + carValue + "]";

}

}

**HomeLoan.java  
  
package** hex.loan.entity;

**public** **class** HomeLoan **extends** Loan {

**private** String propertyAddress;

**private** **int** propertyValue;

**public** HomeLoan() {

**super**();

}

**public** HomeLoan(**int** loanId, Customer customer, **double** principalAmount, **double** interestRate,

**int** loanTerm, String loanType, String loanStatus, String propertyAddress, **int** propertyValue) {

**super**(loanId, customer, principalAmount, interestRate, loanTerm, loanType, loanStatus);

**this**.propertyAddress = propertyAddress;

**this**.propertyValue = propertyValue;

}

**public** String getPropertyAddress() {

**return** propertyAddress;

}

**public** **void** setPropertyAddress(String propertyAddress) {

**this**.propertyAddress = propertyAddress;

}

**public** **int** getPropertyValue() {

**return** propertyValue;

}

**public** **void** setPropertyValue(**int** propertyValue) {

**this**.propertyValue = propertyValue;

}

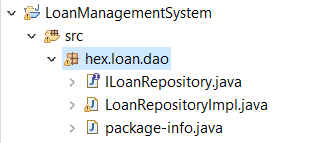
@Override

**public** String toString() {

**return** **super**.toString() + ", HomeLoan [PropertyAddress=" + propertyAddress + ", PropertyValue=" + propertyValue + "]";

}

}

**3. hex.loan.dao**  
  
  
**ILoanRepository.java**  
  
**package** hex.loan.dao;

**import** hex.loan.entity.Loan;

**import** hex.loan.exception.InvalidLoanException;

**import** java.util.List;

**public** **interface** ILoanRepository {

// a. Apply Loan with user confirmation

**void** applyLoan(Loan loan);

// b. Calculate Interest from DB

**double** calculateInterest(**int** loanId) **throws** InvalidLoanException;

// b.i Overloaded - calculate interest using parameters

**double** calculateInterest(**double** principalAmount, **double** interestRate, **int** loanTerm);

// c. Approve/Reject based on credit score

String loanStatus(**int** loanId) **throws** InvalidLoanException;

// d. Calculate EMI from DB

**double** calculateEMI(**int** loanId) **throws** InvalidLoanException;

// d.i Overloaded - calculate EMI using parameters

**double** calculateEMI(**double** principalAmount, **double** annualRate, **int** tenureInMonths);

// e. Repay Loan using a lump sum amount

String loanRepayment(**int** loanId, **double** amount) **throws** InvalidLoanException;

// f. Get All Loans

List<Loan> getAllLoan();

// g. Get Loan By ID

Loan getLoanById(**int** loanId) **throws** InvalidLoanException;

}

**LoanRepositoryImpl.java**  
  
**package** hex.loan.dao;

**import** java.sql.\*;

**import** java.util.\*;

**import** hex.loan.entity.\*;

**import** hex.loan.util.DBConnUtil;

**import** hex.loan.exception.InvalidLoanException;

**public** **class** LoanRepositoryImpl **implements** ILoanRepository {

@Override

**public** **void** applyLoan(Loan loan) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Do you want to apply for this loan? (Yes/No): ");

String confirm = sc.nextLine();

**if** (!confirm.equalsIgnoreCase("yes")) {

System.***out***.println("Loan application cancelled.\*\*\*");

**return**;

}

String insertCustomerSQL = "INSERT INTO Customer (name, email, phone, address, credit\_score) VALUES (?, ?, ?, ?, ?)";

String insertLoanSQL = "INSERT INTO Loan (customer\_id, principal\_amount, interest\_rate, loan\_term, loan\_type, loan\_status) VALUES (?, ?, ?, ?, ?, ?)";

Connection conn = **null**;

PreparedStatement customerStmt = **null**;

PreparedStatement loanStmt = **null**;

**try** {

conn = DBConnUtil.*getDBConn*();

conn.setAutoCommit(**false**);

// 1. Insert Customer

Customer customer = loan.getCustomer();

customerStmt = conn.prepareStatement(insertCustomerSQL, Statement.***RETURN\_GENERATED\_KEYS***);

customerStmt.setString(1, customer.getName());

customerStmt.setString(2, customer.getEmail());

customerStmt.setString(3, customer.getPhone());

customerStmt.setString(4, customer.getAddress());

customerStmt.setInt(5, customer.getCreditScore());

customerStmt.executeUpdate();

ResultSet rs = customerStmt.getGeneratedKeys();

**int** customerId = -1;

**if** (rs.next()) {

customerId = rs.getInt(1);

}

// 2. Insert Loan

loanStmt = conn.prepareStatement(insertLoanSQL);

loanStmt.setInt(1, customerId);

loanStmt.setDouble(2, loan.getPrincipalAmount());

loanStmt.setDouble(3, loan.getInterestRate());

loanStmt.setInt(4, loan.getLoanTerm());

loanStmt.setString(5, loan.getLoanType());

loanStmt.setString(6, loan.getLoanStatus());

loanStmt.executeUpdate();

conn.commit();

System.***out***.println("Loan applied successfully and saved to database.");

} **catch** (SQLException e) {

**try** { **if** (conn != **null**) conn.rollback(); } **catch** (SQLException ex) { ex.printStackTrace(); }

e.printStackTrace();

} **finally** {

**try** {

**if** (customerStmt != **null**) customerStmt.close();

**if** (loanStmt != **null**) loanStmt.close();

**if** (conn != **null**) conn.close();

} **catch** (SQLException ex) { ex.printStackTrace(); }

}

}

@Override

**public** **double** calculateInterest(**int** loanId) **throws** InvalidLoanException {

String sql = "SELECT principal\_amount, interest\_rate, loan\_term FROM Loan WHERE loan\_id = ?";

**try** (Connection conn = DBConnUtil.*getDBConn*();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, loanId);

ResultSet rs = stmt.executeQuery();

**if** (rs.next()) {

**double** p = rs.getDouble("principal\_amount");

**double** r = rs.getDouble("interest\_rate");

**int** n = rs.getInt("loan\_term");

**return** calculateInterest(p, r, n);

} **else** {

**throw** **new** InvalidLoanException("Loan ID not found.");

}

} **catch** (SQLException e) {

e.printStackTrace();

**throw** **new** InvalidLoanException("Database error occurred.");

}

}

@Override

**public** **double** calculateInterest(**double** principalAmount, **double** interestRate, **int** loanTerm) {

**return** (principalAmount \* interestRate \* loanTerm) / 12;

}

@Override

**public** String loanStatus(**int** loanId) **throws** InvalidLoanException {

String selectSQL = "SELECT l.loan\_id, c.credit\_score FROM Loan l JOIN Customer c ON l.customer\_id = c.customer\_id WHERE l.loan\_id = ?";

String updateSQL = "UPDATE Loan SET loan\_status = ? WHERE loan\_id = ?";

**try** (Connection conn = DBConnUtil.*getDBConn*();

PreparedStatement selectStmt = conn.prepareStatement(selectSQL);

PreparedStatement updateStmt = conn.prepareStatement(updateSQL)) {

selectStmt.setInt(1, loanId);

ResultSet rs = selectStmt.executeQuery();

**if** (rs.next()) {

**int** creditScore = rs.getInt("credit\_score");

String newStatus = creditScore > 650 ? "Approved" : "Pending";

updateStmt.setString(1, newStatus);

updateStmt.setInt(2, loanId);

updateStmt.executeUpdate();

**return** "Loan Status Updated: " + newStatus;

} **else** {

**throw** **new** InvalidLoanException("Loan not found with ID: " + loanId);

}

} **catch** (SQLException e) {

e.printStackTrace();

**throw** **new** InvalidLoanException("Error checking loan status.");

}

}

@Override

**public** **double** calculateEMI(**int** loanId) **throws** InvalidLoanException {

String sql = "SELECT principal\_amount, interest\_rate, loan\_term FROM Loan WHERE loan\_id = ?";

**try** (Connection conn = DBConnUtil.*getDBConn*();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, loanId);

ResultSet rs = stmt.executeQuery();

**if** (rs.next()) {

**double** p = rs.getDouble("principal\_amount");

**double** annualRate = rs.getDouble("interest\_rate");

**int** n = rs.getInt("loan\_term");

**return** calculateEMI(p, annualRate, n);

} **else** {

**throw** **new** InvalidLoanException("Loan not found.");

}

} **catch** (SQLException e) {

e.printStackTrace();

**throw** **new** InvalidLoanException("Error calculating EMI.");

}

}

@Override

**public** **double** calculateEMI(**double** principalAmount, **double** annualRate, **int** tenureInMonths) {

**double** r = annualRate / 12 / 100;

**int** n = tenureInMonths;

**return** (principalAmount \* r \* Math.*pow*(1 + r, n)) / (Math.*pow*(1 + r, n) - 1);

}

@Override

**public** String loanRepayment(**int** loanId, **double** amount) **throws** InvalidLoanException {

**double** emi = calculateEMI(loanId);

**if** (amount < emi) {

**return** "Repayment failed: Amount less than single EMI.";

}

**int** paidEmis = (**int**)(amount / emi);

**return** "Repayment success: " + paidEmis + " EMI(s) paid.";

}

@Override

**public** List<Loan> getAllLoan() {

List<Loan> loans = **new** ArrayList<>();

String sql = "SELECT l.\*, c.\* FROM Loan l JOIN Customer c ON l.customer\_id = c.customer\_id";

**try** (Connection conn = DBConnUtil.*getDBConn*();

PreparedStatement stmt = conn.prepareStatement(sql);

ResultSet rs = stmt.executeQuery()) {

**while** (rs.next()) {

Customer c = **new** Customer(

rs.getInt("customer\_id"),

rs.getString("name"),

rs.getString("email"),

rs.getString("phone"),

rs.getString("address"),

rs.getInt("credit\_score")

);

Loan loan = **new** Loan(

rs.getInt("loan\_id"),

c,

rs.getDouble("principal\_amount"),

rs.getDouble("interest\_rate"),

rs.getInt("loan\_term"),

rs.getString("loan\_type"),

rs.getString("loan\_status")

) {}; // Anonymous subclass since Loan is abstract

loans.add(loan);

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** loans;

}

@Override

**public** Loan getLoanById(**int** loanId) **throws** InvalidLoanException {

String sql = "SELECT l.\*, c.\* FROM Loan l JOIN Customer c ON l.customer\_id = c.customer\_id WHERE l.loan\_id = ?";

**try** (Connection conn = DBConnUtil.*getDBConn*();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, loanId);

ResultSet rs = stmt.executeQuery();

**if** (rs.next()) {

Customer c = **new** Customer(

rs.getInt("customer\_id"),

rs.getString("name"),

rs.getString("email"),

rs.getString("phone"),

rs.getString("address"),

rs.getInt("credit\_score")

);

**return** **new** Loan(

rs.getInt("loan\_id"),

c,

rs.getDouble("principal\_amount"),

rs.getDouble("interest\_rate"),

rs.getInt("loan\_term"),

rs.getString("loan\_type"),

rs.getString("loan\_status")

) {};

} **else** {

**throw** **new** InvalidLoanException("Loan ID not found.");

}

} **catch** (SQLException e) {

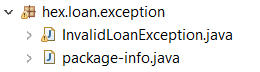
e.printStackTrace();

**throw** **new** InvalidLoanException("Error retrieving loan.");

}

}

}

**4. hex.loan.exception**  
  
**InvalidLoanException.java**  
  
**package** hex.loan.exception;

**public** **class** InvalidLoanException **extends** Exception {

**public** InvalidLoanException() {

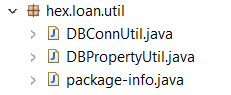
**super**("Invalid loan operation.");

}

**public** InvalidLoanException(String message) {

**super**(message);

}

} **5. hex.loan.util**  
 **DBConnUtil.java**  
 **package** hex.loan.util;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.SQLException;

**import** java.util.Properties;

**public** **class** DBConnUtil {

**public** **static** Connection getDBConn() {

Connection conn = **null**;

**try** {

Properties props = DBPropertyUtil.*getPropertyString*("db/db.properties");

String driver = props.getProperty("driver");

String url = props.getProperty("url");

String username = props.getProperty("username");

String password = props.getProperty("password");

Class.*forName*(driver);

conn = DriverManager.*getConnection*(url, username, password);

} **catch** (ClassNotFoundException | SQLException e) {

System.***out***.println("Database connection failed.");

e.printStackTrace();

}

**return** conn;

}

}

**DBPropertyUtil.java**  
 **package** hex.loan.util;

**import** java.io.FileInputStream;

**import** java.io.IOException;

**import** java.util.Properties;

**public** **class** DBPropertyUtil {

**public** **static** Properties getPropertyString(String filePath) {

Properties props = **new** Properties();

**try** (FileInputStream fis = **new** FileInputStream(filePath)) {

props.load(fis);

} **catch** (IOException e) {

System.***out***.println("Error loading db.properties: " + e.getMessage());

}

**return** props;

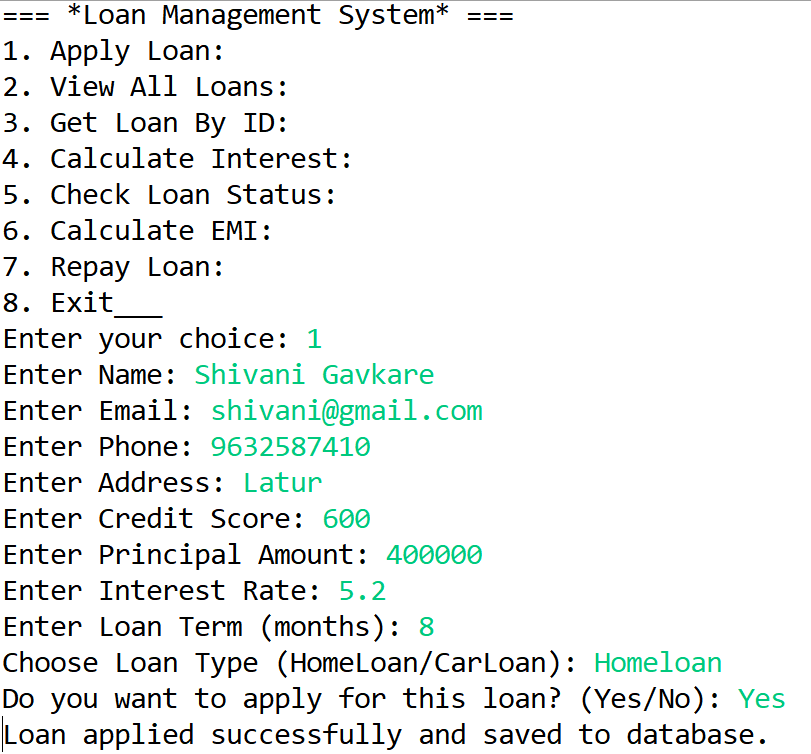
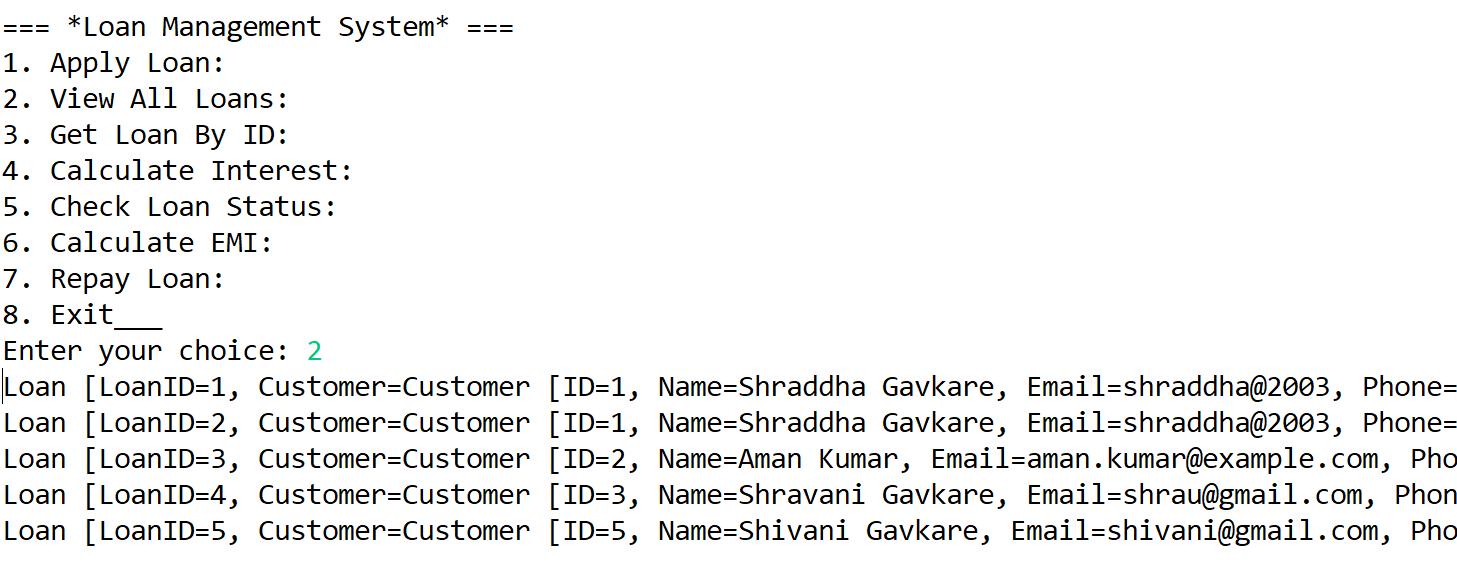
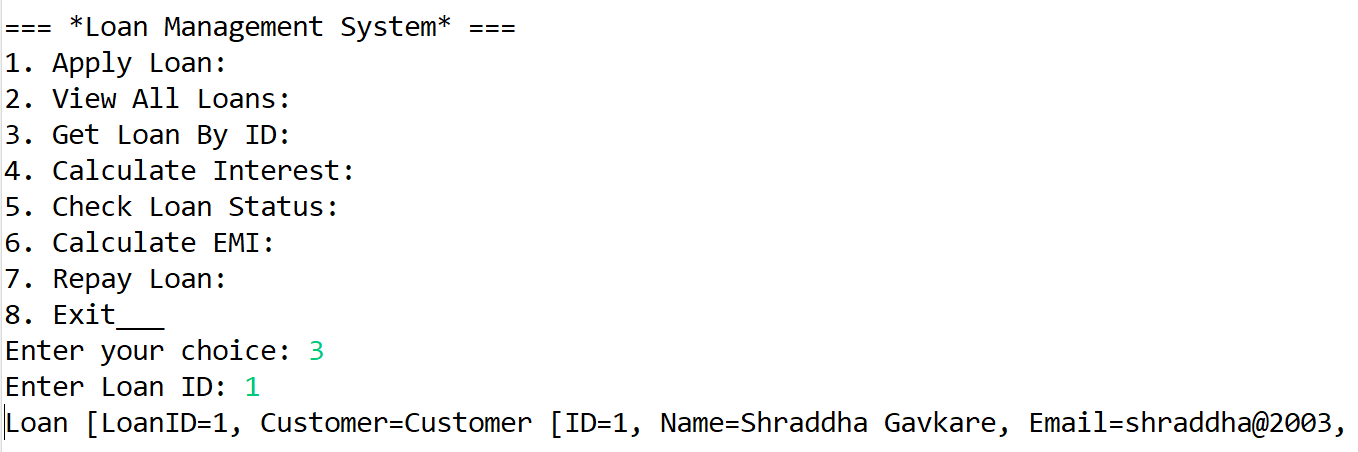
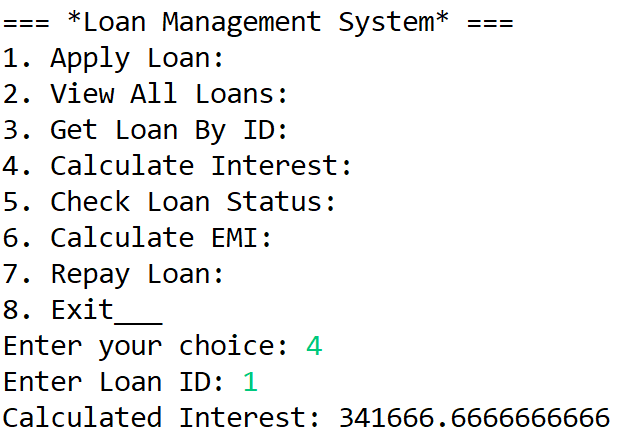
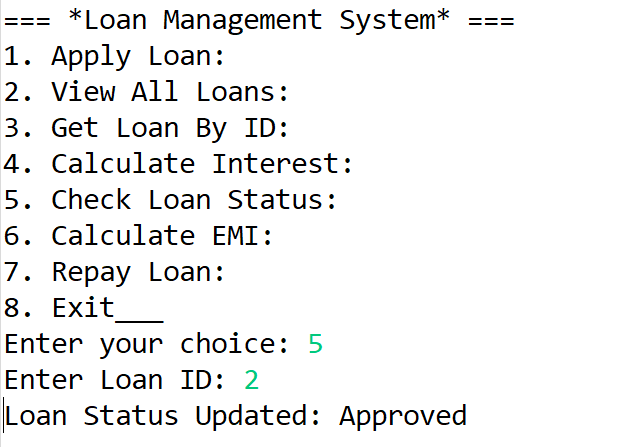
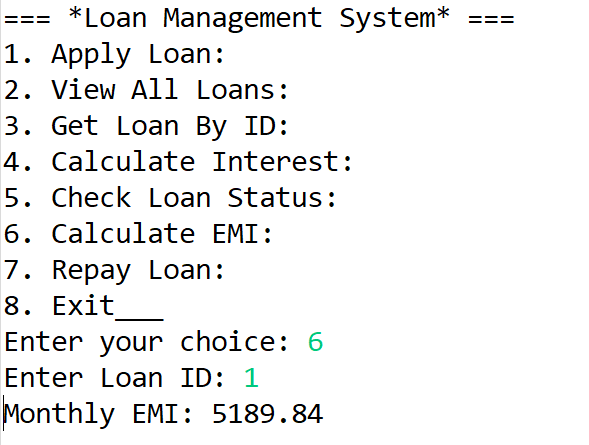
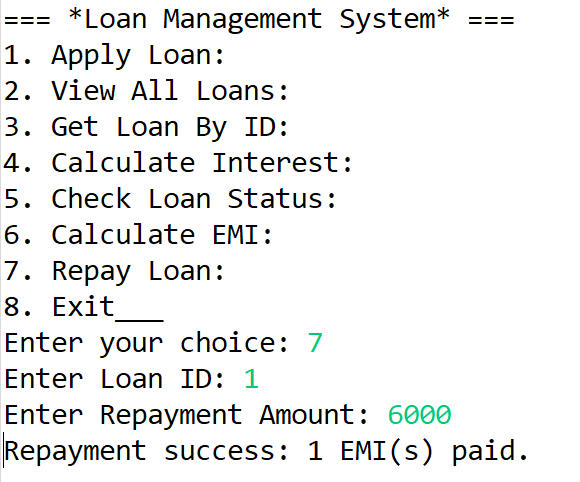
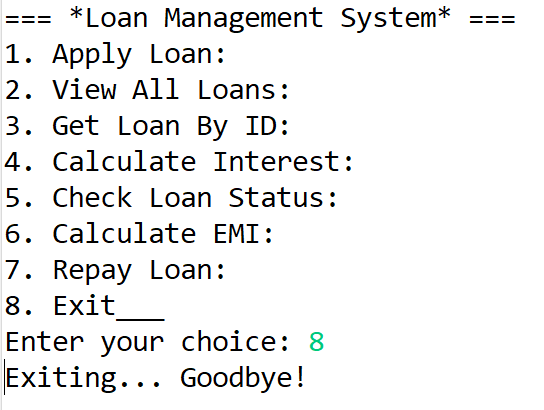
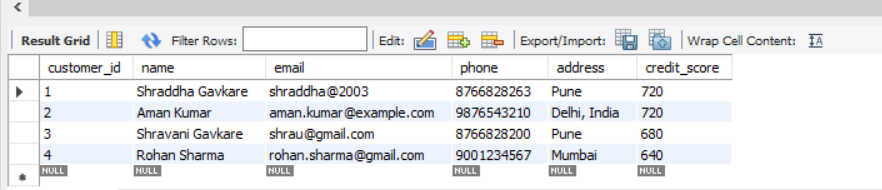
}

} **Db.properties**driver=com.mysql.cj.jdbc.Driver

url=jdbc:mysql://localhost:3306/LoanDB

username=root

password=Shraddha@2003

**5. Output Screens / Terminal Results**  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**Database tables:  
  
1) Customer table:**  
**2) Loan table:**  
