# **Project report**



**ACE ENGINEERING COLLEGE, GATHKESAR**

***(An Autonomous Institute under JNTU, Hyderabad)***

**INFORMATION TECHNOLOGY**

**Title:** Telecom Billing System

**Presented By**

Kokala Srinavya (21AG1A1228)

Kondam Pooja (21AG1A1230)

Sindhuja sunkari (21AG1A1250)

**GUIDE BY**: KRISHNANAD

CONTENTS

* Abstract
* Introduction
* Functional Requirement
* ER Diagram
* Queries To Create DataBase
* Class/UML Attributes
* All Class Java File Codes
* Challenges List
* Project Packages

**Abstract**

The Telecom Billing System (TBS) is a crucial component of modern telecommunication networks, facilitating the accurate and efficient management of subscriber accounts, usage tracking, and billing processes.

The primary objective of the TBS project is to design and develop a comprehensive billing system tailored to the specific needs of telecommunication service providers. The system aims to automate billing processes, improve billing accuracy, enhance revenue management, and streamline customer service operations.

Implementation of the TBS involves coding, testing, and integration of software components, ensuring scalability, reliability, and security. The system utilizes database technologies for data storage and retrieval, with robust security measures to protect sensitive subscriber information and financial transactions.

In conclusion, the Telecom Billing System represents a significant advancement in billing and revenue management for telecommunication service providers. Its successful implementation promises tangible benefits in terms of operational efficiency, revenue optimization, and customer satisfaction, positioning service providers for sustained growth and success in the dynamic telecommunications industry.

**Introduction**

A telecom billing system is the backbone of any telecommunications service provider, facilitating the complex process of tracking, managing, and invoicing for the usage of their services. From mobile phone calls to internet data usage, every interaction with the network generates data that needs to be accuratelrecorded, processed, and billed.

Telecom billing systems are sophisticated software platforms that handle this task seamlessly, ensuring that customers are billed correctly for the services they use. These systems typically integrate with various network elements to collect usage data, apply appropriate tariffs or pricing plans, generate invoices, and manage payments.

In addition to billing for traditional services like voice calls and SMS, modern telecom billing systems also handle billing for a wide range of digital services, including mobile data, content subscriptions, and value-added services.

**Functional Requirements**

1. **Usage Tracking:** The system should be able to accurately track usage data for various telecom services such as voice calls, SMS, MMS, data usage, roaming, and value-added services.

2. **Tariff Management:** It should support the configuration and management of different tariffs, pricing plans, discounts, and promotions based on factors like time of day, destination, service type, and customer segment.

3. **Customer Management:** The system should allow for the creation, modification, and management of customer accounts, including personal information, subscription details, billing preferences, and payment methods.

4. **Rating and Charging:** It should be capable of rating usage records according to the applicable tariffs and pricing plans, calculating charges accurately based on usage volume, duration, distance, or other relevant parameters.

5. **Invoicing and Billing:** The system should generate itemized invoices or bills for customers, consolidating charges for all consumed services during a billing cycle. Invoices should be customizable and include detailed usage information, taxes, discounts, and any other applicable fees.

6. **Payment Processing:** It should support various payment methods such as credit/debit cards, bank transfers, mobile wallets, and direct debits. The system should handle payment processing securely and provide options for automated payments, payment reminders, and overdue notifications.

7. **Account Reconciliation**: The system should reconcile billing records with payment transactions to ensure accuracy and integrity in financial reporting. It should also handle adjustments, refunds, and dispute resolution processes efficiently.

8. **Reporting and Analytics**: The system should offer comprehensive reporting and analytics capabilities, allowing telecom operators to analyze usage trends, revenue streams, customer behavior, and profitability. Reports should be customizable, intuitive, and accessible to authorized users.

9. **Integration and Interoperability**: It should integrate seamlessly with other systems and network elements such as CRM (Customer Relationship Management) systems, provisioning systems, network elements, and third-party billing systems for wholesale or interconnect billing.

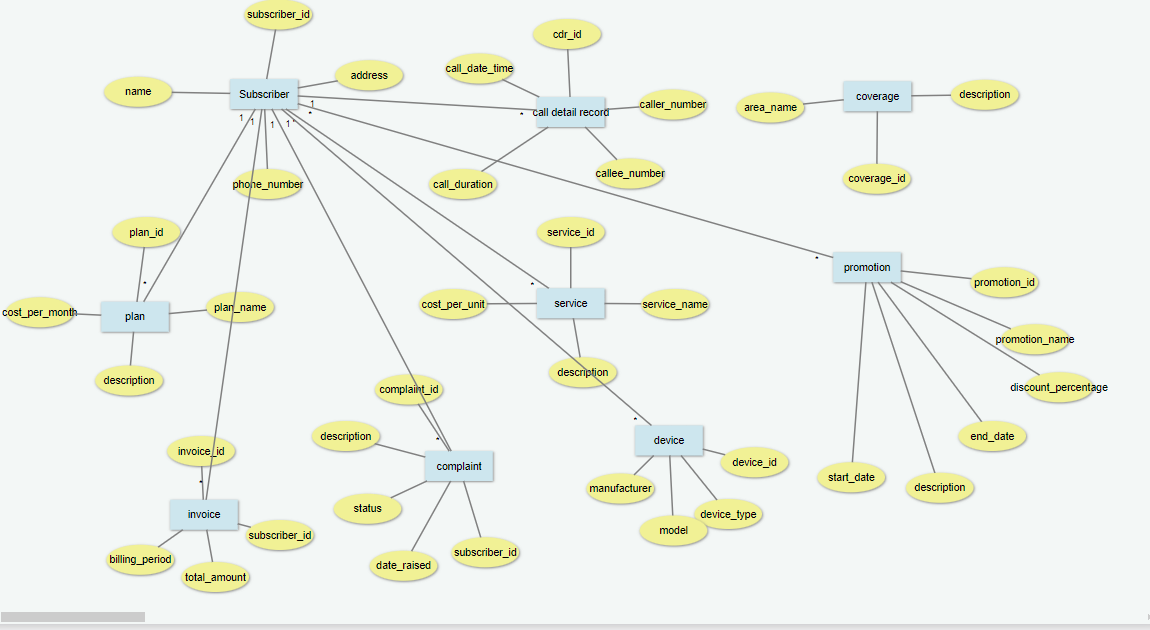
10. **Compliance and Regulatory Support**: The system should comply with industry regulations, data protection laws, and taxation requirements applicable to telecom services in various jurisdictions. It should also support compliance reporting and audit trails.

11. **Scalability and Performance**: The system should be scalable to accommodate growth in subscriber base and usage volume. It should be capable of handling peak loads efficiently while maintaining high performance and reliability.

12. **User Interface and Accessibility:** The system should have a user-friendly interface for both administrative users and customers, supporting multi-channel access (web, mobile app, IVR) and providing self-service capabilities for account management, bill inquiries, and service activations.

These functional requirements form the foundation of a comprehensive telecom billing system, enabling telecom operators to manage their billing processes effectively and deliver a seamless customer experience.

**ER DIAGRAM**



**QUERIES**

CREATE TABLE Subscriber (

SubscriberID INT PRIMARY KEY AUTO\_INCREMENT,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE,

Gender ENUM('Male', 'Female', 'Other'),

Email VARCHAR(100),

PhoneNumber VARCHAR(15) UNIQUE,

Address VARCHAR(255));

CREATE TABLE Plan (

PlanID INT PRIMARY KEY,

Name VARCHAR(255),

Cost DECIMAL(10,2)

); CREATE TABLE call\_detail\_record (

cdr\_id SERIAL PRIMARY KEY,

caller\_id VARCHAR(15) NOT NULL,

callee\_id VARCHAR(15) NOT NULL,

start\_time TIMESTAMP NOT NULL,

end\_time TIMESTAMP NOT NULL,

duration INT NOT NULL,

call\_type VARCHAR(10) NOT NULL,

call\_cost DECIMAL(10, 2) NOT NULL,

call\_status VARCHAR(20) NOT NULL,

tariff\_plan VARCHAR(50)

); CREATE TABLE Service (

ServiceID INT PRIMARY KEY,

Name VARCHAR(255),

PlanID INT,

FOREIGN KEY (PlanID) REFERENCES Plan(PlanID)

);

CREATE TABLE Device (

DeviceID INT PRIMARY KEY AUTO\_INCREMENT,

SubscriberID INT NOT NULL,

DeviceType VARCHAR(50) NOT NULL,

Model VARCHAR(100) NOT NULL,

Manufacturer VARCHAR(100) NOT NULL,

PurchaseDate DATE NOT NULL,

Status VARCHAR(20) NOT NULL CHECK (Status IN ('Active', 'Inactive', 'Lost', 'Damaged')),

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UpdatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (SubscriberID) REFERENCES Subscriber(SubscriberID)

);

CREATE TABLE Complaint (

ComplaintID INT PRIMARY KEY AUTO\_INCREMENT,

SubscriberID INT NOT NULL,

Description TEXT NOT NULL,

Status VARCHAR(20) NOT NULL CHECK (Status IN ('Open', 'In Progress', 'Resolved', 'Closed')),

DateRaised DATE NOT NULL,

DateResolved DATE,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UpdatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (SubscriberID) REFERENCES Subscriber(SubscriberID)

);

CREATE TABLE Coverage (

CoverageID INT PRIMARY KEY AUTO\_INCREMENT,

AreaName VARCHAR(100) NOT NULL,

Description TEXT,

Latitude DECIMAL(10, 7),

Longitude DECIMAL(10, 7),

ServiceType VARCHAR(50) NOT NULL,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UpdatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

); CREATE TABLE Promotion (

promotion\_id INT PRIMARY KEY,

promotion\_name VARCHAR(100) NOT NULL,

start\_date DATE NOT NULL,

end\_date DATE NOT NULL,

description TEXT,

discount DECIMAL(5, 2),

status VARCHAR(20) NOT NULL

);

CREATE TABLE Customer (

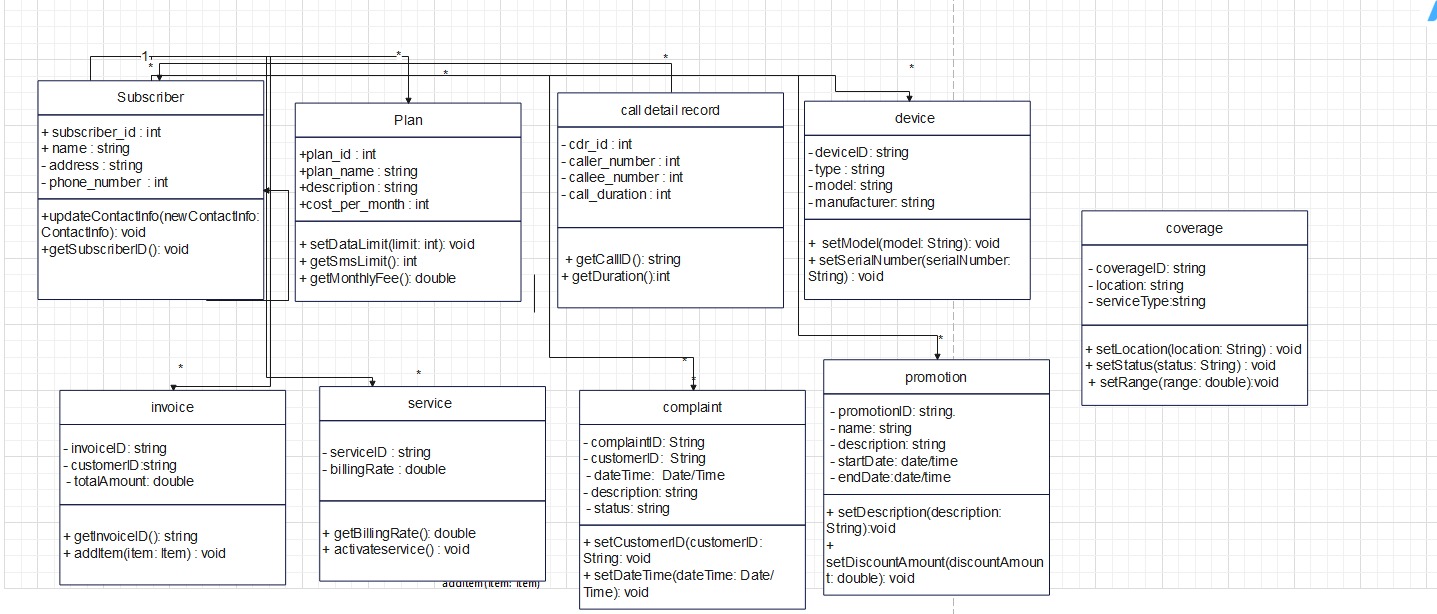
CustomerID INT PRIMARY KEY,

Name VARCHAR(255),

Address VARCHAR(255)

);

**UML DIAGRAM**



**ALL JAVA FILE CODES (ALL CLASSES):**

**1.Subscriber:**

public class Subscriber {

private int subscriberId;

private String phoneNumber;

private String name;

private String address;

private double balance;

// Constructors

public Subscriber() {}

public Subscriber(int subscriberId, String phoneNumber, String name, String address, double balance) {

this.subscriberId = subscriberId;

this.phoneNumber = phoneNumber;

this.name = name;

this.address = address;

this.balance = balance;

}

// Getters

public int getSubscriberId() {

return subscriberId;

}

public String getPhoneNumber() {

return phoneNumber;

}

public String getName() {

return name;

}

public String getAddress() {

return address;

}

public double getBalance() {

return balance;

}

// Setters

public void setSubscriberId(int subscriberId) {

this.subscriberId = subscriberId;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber = phoneNumber;

}

public void setName(String name) {

this.name = name;

}

public void setAddress(String address) {

this.address = address;

}

public void setBalance(double balance) {

this.balance = balance;

}

// Display method

public void displayDetails() {

System.out.println("Subscriber ID: " + subscriberId);

System.out.println("Phone Number: " + phoneNumber);

System.out.println("Name: " + name);

System.out.println("Address: " + address);

System.out.println("Balance: $" + balance);

System.out.println();

}

// Method to recharge balance

public void rechargeBalance(double amount) {

balance += amount;

System.out.println("Balance recharged successfully. Current balance: $" + balance);

}

// Method to deduct balance for call

public void makeCall(double callDuration) {

double callCost = callDuration \* 0.1; // Assuming call cost per minute is $0.1

if (balance >= callCost) {

balance -= callCost;

System.out.println("Call made successfully. Call cost: $" + callCost + ", Remaining balance: $" + balance);

} else {

System.out.println("Insufficient balance to make call.");

}

}

}

2.invoice:

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

public class Invoice {

private int invoiceId;

private int subscriberId;

private Date startDate;

private Date endDate;

private double totalAmount;

private List<CallDetailRecord> callDetailRecords;

// Constructors

public Invoice() {

callDetailRecords = new ArrayList<>();

}

public Invoice(int invoiceId, int subscriberId, Date startDate, Date endDate) {

this.invoiceId = invoiceId;

this.subscriberId = subscriberId;

this.startDate = startDate;

this.endDate = endDate;

this.totalAmount = 0.0;

callDetailRecords = new ArrayList<>();

}

// Getters

public int getInvoiceId() {

return invoiceId;

}

public int getSubscriberId() {

return subscriberId;

}

public Date getStartDate() {

return startDate;

}

public Date getEndDate() {

return endDate;

}

public double getTotalAmount() {

return totalAmount;

}

public List<CallDetailRecord> getCallDetailRecords() {

return callDetailRecords;

}

// Setters

public void setInvoiceId(int invoiceId) {

this.invoiceId = invoiceId;

}

public void setSubscriberId(int subscriberId) {

this.subscriberId = subscriberId;

}

public void setStartDate(Date startDate) {

this.startDate = startDate;

}

public void setEndDate(Date endDate) {

this.endDate = endDate;

}

public void setTotalAmount(double totalAmount) {

this.totalAmount = totalAmount;

}

// Add a CallDetailRecord to the invoice

public void addCallDetailRecord(CallDetailRecord cdr) {

callDetailRecords.add(cdr);

}

// Display method

public void display() {

System.out.println("Invoice Details:");

System.out.println("Invoice ID: " + invoiceId);

System.out.println("Subscriber ID: " + subscriberId);

System.out.println("Start Date: " + startDate);

System.out.println("End Date: " + endDate);

System.out.println("Total Amount: $" + totalAmount);

System.out.println("Call Detail Records:");

for (CallDetailRecord cdr : callDetailRecords) {

cdr.display();

}

}

}

**3.Device:**

public class Device {

private int deviceId;

private String serialNumber;

private String model;

private String manufacturer;

// Constructors

public Device() {}

public Device(int deviceId, String serialNumber, String model, String manufacturer) {

this.deviceId = deviceId;

this.serialNumber = serialNumber;

this.model = model;

this.manufacturer = manufacturer;

}

// Getters

public int getDeviceId() {

return deviceId;

}

public String getSerialNumber() {

return serialNumber;

}

public String getModel() {

return model;

}

public String getManufacturer() {

return manufacturer;

}

// Setters

public void setDeviceId(int deviceId) {

this.deviceId = deviceId;

}

public void setSerialNumber(String serialNumber) {

this.serialNumber = serialNumber;

}

public void setModel(String model) {

this.model = model;

}

public void setManufacturer(String manufacturer) {

this.manufacturer = manufacturer;

}

// Display method

public void display() {

System.out.println("Device Details:");

System.out.println("Device ID: " + deviceId);

System.out.println("Serial Number: " + serialNumber);

System.out.println("Model: " + model);

System.out.println("Manufacturer: " + manufacturer);

System.out.println();

}

}

**4.Complaint:**

public class Complaint {

private int complaintId;

private int subscriberId;

private String description;

private boolean resolved;

// Constructors

public Complaint() {}

public Complaint(int complaintId, int subscriberId, String description, boolean resolved) {

this.complaintId = complaintId;

this.subscriberId = subscriberId;

this.description = description;

this.resolved = resolved;

}

// Getters

public int getComplaintId() {

return complaintId;

}

public int getSubscriberId() {

return subscriberId;

}

public String getDescription() {

return description;

}

public boolean isResolved() {

return resolved;

}

// Setters

public void setComplaintId(int complaintId) {

this.complaintId = complaintId;

}

public void setSubscriberId(int subscriberId) {

this.subscriberId = subscriberId;

}

public void setDescription(String description) {

this.description = description;

}

public void setResolved(boolean resolved) {

this.resolved = resolved;

}

// Display method

public void display() {

System.out.println("Complaint Details:");

System.out.println("Complaint ID: " + complaintId);

System.out.println("Subscriber ID: " + subscriberId);

System.out.println("Description: " + description);

System.out.println("Resolved: " + (resolved ? "Yes" : "No"));

System.out.println();

}

}

**5.Coverage:**

public class Coverage {

private int coverageId;

private String area;

private boolean isCovered;

// Constructors

public Coverage() {}

public Coverage(int coverageId, String area, boolean isCovered) {

this.coverageId = coverageId;

this.area = area;

this.isCovered = isCovered;

}

// Getters

public int getCoverageId() {

return coverageId;

}

public String getArea() {

return area;

}

public boolean isCovered() {

return isCovered;

}

// Setters

public void setCoverageId(int coverageId) {

this.coverageId = coverageId;

}

public void setArea(String area) {

this.area = area;

}

public void setCovered(boolean covered) {

isCovered = covered;

}

// Display method

public void display() {

System.out.println("Coverage Details:");

System.out.println("Coverage ID: " + coverageId);

System.out.println("Area: " + area);

System.out.println("Is Covered: " + (isCovered ? "Yes" : "No"));

System.out.println();

}

}

**6.Promotion:**

import java.util.Date;

public class Promotion {

private int promotionId;

private String name;

private String description;

private Date startDate;

private Date endDate;

private double discountPercentage;

// Constructors

public Promotion() {}

public Promotion(int promotionId, String name, String description, Date startDate, Date endDate, double discountPercentage) {

this.promotionId = promotionId;

this.name = name;

this.description = description;

this.startDate = startDate;

this.endDate = endDate;

this.discountPercentage = discountPercentage;

}

// Getters

public int getPromotionId() {

return promotionId;

}

public String getName() {

return name;

}

public String getDescription() {

return description;

}

public Date getStartDate() {

return startDate;

}

public Date getEndDate() {

return endDate;

}

public double getDiscountPercentage() {

return discountPercentage;

}

// Setters

public void setPromotionId(int promotionId) {

this.promotionId = promotionId;

}

public void setName(String name) {

this.name = name;

}

public void setDescription(String description) {

this.description = description;

}

public void setStartDate(Date startDate) {

this.startDate = startDate;

}

public void setEndDate(Date endDate) {

this.endDate = endDate;

}

public void setDiscountPercentage(double discountPercentage) {

this.discountPercentage = discountPercentage;

}

// Display method

public void display() {

System.out.println("Promotion Details:");

System.out.println("Promotion ID: " + promotionId);

System.out.println("Name: " + name);

System.out.println("Description: " + description);

System.out.println("Start Date: " + startDate);

System.out.println("End Date: " + endDate);

System.out.println("Discount Percentage: " + discountPercentage + "%");

System.out.println();

}

}

**7.CallDetailRecord**:

import java.util.Date;

public class CallDetailRecord {

private int callId;

private int subscriberId;

private String phoneNumber;

private Date startTime;

private Date endTime;

private double duration;

private double cost;

// Constructors

public CallDetailRecord() {}

public CallDetailRecord(int callId, int subscriberId, String phoneNumber, Date startTime, Date endTime, double duration, double cost) {

this.callId = callId;

this.subscriberId = subscriberId;

this.phoneNumber = phoneNumber;

this.startTime = startTime;

this.endTime = endTime;

this.duration = duration;

this.cost = cost;

}

// Getters

public int getCallId() {

return callId;

}

public int getSubscriberId() {

return subscriberId;

}

public String getPhoneNumber() {

return phoneNumber;

}

public Date getStartTime() {

return startTime;

}

public Date getEndTime() {

return endTime;

}

public double getDuration() {

return duration;

}

public double getCost() {

return cost;

}

// Setters

public void setCallId(int callId) {

this.callId = callId;

}

public void setSubscriberId(int subscriberId) {

this.subscriberId = subscriberId;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber = phoneNumber;

}

public void setStartTime(Date startTime) {

this.startTime = startTime;

}

public void setEndTime(Date endTime) {

this.endTime = endTime;

}

public void setDuration(double duration) {

this.duration = duration;

}

public void setCost(double cost) {

this.cost = cost;

}

// Display method

public void display() {

System.out.println("Call Detail Record:");

System.out.println("Call ID: " + callId);

System.out.println("Subscriber ID: " + subscriberId);

System.out.println("Phone Number: " + phoneNumber);

System.out.println("Start Time: " + startTime);

System.out.println("End Time: " + endTime);

System.out.println("Duration (minutes): " + duration);

System.out.println("Cost: $" + cost);

System.out.println();

}

}

**8.Service:**

public class Service {

private int serviceId;

private String name;

private double monthlyFee;

// Constructors

public Service() {}

public Service(int serviceId, String name, double monthlyFee) {

this.serviceId = serviceId;

this.name = name;

this.monthlyFee = monthlyFee;

}

// Getters

public int getServiceId() {

return serviceId;

}

public String getName() {

return name;

}

public double getMonthlyFee() {

return monthlyFee;

}

// Setters

public void setServiceId(int serviceId) {

this.serviceId = serviceId;

}

public void setName(String name) {

this.name = name;

}

public void setMonthlyFee(double monthlyFee) {

this.monthlyFee = monthlyFee;

}

// Display method

public void display() {

System.out.println("Service Details:");

System.out.println("Service ID: " + serviceId);

System.out.println("Name: " + name);

System.out.println("Monthly Fee: $" + monthlyFee);

System.out.println();

}

}

**9.PLAN:**

public class Plan {

private int planId;

private String name;

private double monthlyFee;

private int includedMinutes;

private double minuteRate;

// Constructors

public Plan() {}

public Plan(int planId, String name, double monthlyFee, int includedMinutes, double minuteRate) {

this.planId = planId;

this.name = name;

this.monthlyFee = monthlyFee;

this.includedMinutes = includedMinutes;

this.minuteRate = minuteRate;

}

// Getters

public int getPlanId() {

return planId;

}

public String getName() {

return name;

}

public double getMonthlyFee() {

return monthlyFee;

}

public int getIncludedMinutes() {

return includedMinutes;

}

public double getMinuteRate() {

return minuteRate;

}

// Setters

public void setPlanId(int planId) {

this.planId = planId;

}

public void setName(String name) {

this.name = name;

}

public void setMonthlyFee(double monthlyFee) {

this.monthlyFee = monthlyFee;

}

public void setIncludedMinutes(int includedMinutes) {

this.includedMinutes = includedMinutes;

}

public void setMinuteRate(double minuteRate) {

this.minuteRate = minuteRate;

}

// Display method

public void display() {

System.out.println("Plan Details:");

System.out.println("Plan ID: " + planId);

System.out.println("Name: " + name);

System.out.println("Monthly Fee: $" + monthlyFee);

System.out.println("Included Minutes: " + includedMinutes);

System.out.println("Minute Rate: $" + minuteRate);

System.out.println();

}

}

**10.PAYMENT:**

import java.util.Date;

public class Payment {

private int paymentId;

private int subscriberId;

private Date paymentDate;

private double amount;

// Constructors

public Payment() {}

public Payment(int paymentId, int subscriberId, Date paymentDate, double amount) {

this.paymentId = paymentId;

this.subscriberId = subscriberId;

this.paymentDate = paymentDate;

this.amount = amount;

}

// Getters

public int getPaymentId() {

return paymentId;

}

public int getSubscriberId() {

return subscriberId;

}

public Date getPaymentDate() {

return paymentDate;

}

public double getAmount() {

return amount;

}

// Setters

public void setPaymentId(int paymentId) {

this.paymentId = paymentId;

}

public void setSubscriberId(int subscriberId) {

this.subscriberId = subscriberId;

}

public void setPaymentDate(Date paymentDate) {

this.paymentDate = paymentDate;

}

public void setAmount(double amount) {

this.amount = amount;

}

// Display method

public void display() {

System.out.println("Payment Details:");

System.out.println("Payment ID: " + paymentId);

System.out.println("Subscriber ID: " + subscriberId);

System.out.println("Payment Date: " + paymentDate);

System.out.println("Amount: $" + amount);

System.out.println();

}

}

**MAIN METHOD:**

package fooddelivery;

import java.util.Date;

public class Main {

public static void main(String[] args) {

// Creating Subscriber instances

Subscriber subscriber1 = new Subscriber(1, "123-456-7890", "John Doe", "123 Main St", 50.00);

Subscriber subscriber2 = new Subscriber(2, "987-654-3210", "Jane Smith", "456 Elm St", 75.00);

// Displaying Subscriber details

subscriber1.displayDetails();

subscriber2.displayDetails();

// Recharging balance for subscriber1

subscriber1.rechargeBalance(25.00);

// Making a call from subscriber2

subscriber2.makeCall(30); // 30 minutes call

// Creating CallDetailRecord instances

CallDetailRecord cdr1 = new CallDetailRecord(1, 1, new Date(), 10);

CallDetailRecord cdr2 = new CallDetailRecord(2, 1, new Date(), 20);

CallDetailRecord cdr3 = new CallDetailRecord(3, 2, new Date(), 30);

// Creating Invoice instances

Invoice invoice1 = new Invoice(1, 1, new Date(), new Date());

Invoice invoice2 = new Invoice(2, 2, new Date(), new Date());

// Adding CallDetailRecords to invoices

invoice1.addCallDetailRecord(cdr1);

invoice1.addCallDetailRecord(cdr2);

invoice2.addCallDetailRecord(cdr3);

// Displaying Invoice details

invoice1.display();

invoice2.display();

}

}

class CallDetailRecord {

private int callId;

private int subscriberId;

private Date callDate;

private int duration; // Duration in minutes

// Constructor

public CallDetailRecord(int callId, int subscriberId, Date callDate, int duration) {

this.callId = callId;

this.subscriberId = subscriberId;

this.callDate = callDate;

this.duration = duration;

}

// Getters and Setters

public int getCallId() {

return callId;

}

public void setCallId(int callId) {

this.callId = callId;

}

public int getSubscriberId() {

return subscriberId;

}

public void setSubscriberId(int subscriberId) {

this.subscriberId = subscriberId;

}

public Date getCallDate() {

return callDate;

}

public void setCallDate(Date callDate) {

this.callDate = callDate;

}

public int getDuration() {

return duration;

}

public void setDuration(int duration) {

this.duration = duration;

}

// Display method

public void display() {

System.out.println("Call Detail Record:");

System.out.println("Call ID: " + callId);

System.out.println("Subscriber ID: " + subscriberId);

System.out.println("Call Date: " + callDate);

System.out.println("Duration: " + duration + " minutes");

System.out.println();

}

}

**Challenges list**

Representing complex relationships, especially many-to-many relationships, can be challenging. These relationships involve multiple entities and require careful modeling to avoid ambiguity.

Creating the tables in a specific order and inserting the values accordingly

Handling sensitive information such as patient health records requires strict adherence to data security and privacy regulations. Ensuring data encryption, access controls, and secure storage mechanisms is crucial.