**public class User** {

// γνωρίσματα - καταστάσεις της κλάσης

private int userId; // μοναδικό αναγνωριστικό χρήστη

private String username; // ονομα χρήστη

private String passwordHash; // κατακερματ. κωδικός πρόσβασης

// associations - Ο user έχει πολλές κρατήσεις

private List<Booking> bookings;

// Constructor

public User(int userId, String username, String passwordHash) {

this.userId = userId;

this.username = username;

this.passwordHash = passwordHash;

this.bookings = new ArrayList<>(); // Αρχικοποίηση της λίστας κρατήσεων

}

// Getters

public int getUserId() {

return userId;

}

public String getUsername() {

return username;

}

//για τις μεθόδους - η λογική θα προστεθεί σε επόμενα βηματα

public boolean authenticate(String username, String password) {

System.out.println("User.authenticate() placeholder called.");

return false; // Προσωρινή επιστροφή

}

public Customer getCustomerProfile() {

System.out.println("User.getCustomerProfile() placeholder called.");

// Προσωρινό: Επιστρέφει ένα dummy Customer, θα συνδεθεί σε επομενα βηματα

return new Customer(this.userId, "placeholder@example.com", "123-456-7890");

}

public List<Booking> getBookings() {

System.out.println("User.getBookings() placeholder called.");

return bookings; // Επιστρέφει την πραγματική λίστα που αρχικοποιήθηκε

}

public void addBooking(Booking booking) {

System.out.println("User.addBooking() placeholder called.");

this.bookings.add(booking); // Προσθήκη στην λίστα

}

}

**public class Customer** {

// γνωρίσματα

private int customerId; // μοναδικό αναγνωριστικό πελάτη

private String email; // εmail πελάτη

private String phoneNumber; // τηλέφωνο πελάτη

private int violationCount; // αριθμός παραβάσεων

// Associations - Ο Customer έχει πολλές κρατήσεις

private List<Booking> bookings;

// Constructor

public Customer(int customerId, String email, String phoneNumber) {

this.customerId = customerId;

this.email = email;

this.phoneNumber = phoneNumber;

this.violationCount = 0; // Αρχική τιμή 0

this.bookings = new ArrayList<>(); // αρχικοποίηση της λίστας

}

// Getters

public int getCustomerId() {

return customerId;

}

public String getEmail() {

return email;

}

public String getPhoneNumber() {

return phoneNumber;

}

public int getViolationCount() {

return violationCount;

}

// για τις μεθόδους - η λογική θα προστεθεί στα επομενα βηματα

public void incrementViolationCount() {

System.out.println("Customer.incrementViolationCount() placeholder called.");

// Η λογική αύξησης θα προστεθεί στα επομενα βηματα

}

public List<Booking> getBookings() {

System.out.println("Customer.getBookings() placeholder called.");

return bookings; // επιστρέφει την πραγματική λίστα

}

public void addBooking(Booking booking) {

System.out.println("Customer.addBooking() placeholder called.");

this.bookings.add(booking); // προσθήκη στην λιστα

}

}

**public class Booking** {

// γνωρίσματα

private int bookingId; // μοναδικό αναγνωριστικό κράτησης

private LocalDateTime date; // ημερομηνια και ώρα κράτησης

private String vehicleType; // τυπος οχήματος

private String pickupLocation; // σημειο παραλαβής

private String dropoffLocation; // σημειο παράδοσης

private String paymentStatus; // κατασταση πληρωμής

private double originalCost; // αρχικό κοστος

private String status; // π.χ. Confirmed, Cancelled

// Constructor

public Booking(int bookingId, LocalDateTime date, String vehicleType,String pickupLocation, String dropoffLocation, double originalCost)

{

this.bookingId = bookingId;

this.date = date;

this.vehicleType = vehicleType;

this.pickupLocation = pickupLocation;

this.dropoffLocation = dropoffLocation;

this.originalCost = originalCost;

this.paymentStatus = "Pending"; // προεπιλεγμενη κατσταση

this.status = "Confirmed"; // προεπιλεγμενη κατσταση

}

// Getters

public int getBookingId() {

return bookingId;

}

public LocalDateTime getDate() {

return date;

}

public String getVehicleType() {

return vehicleType;

}

public String getPickupLocation() {

return pickupLocation;

}

public String getDropoffLocation() {

return dropoffLocation;

}

public String getPaymentStatus() {

return paymentStatus;

}

public double getOriginalCost() {

return originalCost;

}

public String getStatus() {

return status;

}

// Setters

public void setPaymentStatus(String paymentStatus) {

this.paymentStatus = paymentStatus;

}

public void setStatus(String status) {

this.status = status;

}

//για τις μεθόδους - η λογική θα προστεθεί σε επομενα βηματα

public int getRemainingTimeInHours() {

System.out.println("Booking.getRemainingTimeInHours() placeholder called.");

return 0; // προσωρινη επιστροφη

}

public boolean cancel(Cancellation cancellationDetails) {

// Χρειάζεται η κλάση Cancellation

System.out.println("Booking.cancel() placeholder called.");

return false; // προσωρινη επιστροφη

}

}

**public class Cancellation** {

// γνωρσματα

private int cancellationId; // μοναδικό αναγνωριστικό ακύρωσης

private LocalDateTime cancellationDate; // ημερομηνίι/ώρα ακύρωσης

private String cancellationReason; // λόγος ακύρωσης

private double refundAmount; // ποσό επιστροφής

private boolean penaltyApplied; // εχει εφαρμοστει ποινή?

// Constructor

public Cancellation(int cancellationId, String cancellationReason) {

this.cancellationId = cancellationId;

this.cancellationDate = LocalDateTime.now(); //ημ.δημιουργίας της ακύρωσης

this.cancellationReason = cancellationReason;

this.refundAmount = 0.0; // αρχικη τιμη

this.penaltyApplied = false; // αρχικη τιμη

}

// Getters

public int getCancellationId() {

return cancellationId;

}

public LocalDateTime getCancellationDate() {

return cancellationDate;

}

public String getCancellationReason() {

return cancellationReason;

}

public double getRefundAmount() {

return refundAmount;

}

public boolean isPenaltyApplied() {

return penaltyApplied;

}

// Setters

public void setRefundAmount(double refundAmount) {

this.refundAmount = refundAmount;

}

public void setPenaltyApplied(boolean penaltyApplied) {

this.penaltyApplied = penaltyApplied;

}

//για τις μεθόδους - η λογική θα προστεθει σε επομενα βηματα

public double calculateRefund(double originalCost,CancellationPolicy policy, int remainingHours) {

// Χρειάζεται CancellationPolicy

System.out.println("Cancellation.calculateRefund() placeholder called.");

return 0.0; // προσωρινη επιστροφη

}

public double applyPenalty(double originalCost, CancellationPolicy policy) {

System.out.println("Cancellation.applyPenalty() placeholder called.");

return 0.0; // προσωρινη επιστροφη

}

}

**public class CancellationPolicy** {

// γνωρισματα

private int policyId; // μοναδικό αναγνωριστικό πολιτικής

private String policyName; // ονομα πολιτικης

private String ruleSetDescription; // περιγραφή κανόνων

// Constructor

public CancellationPolicy(int policyId, String policyName, String ruleSetDescription) {

this.policyId = policyId;

this.policyName = policyName;

this.ruleSetDescription = ruleSetDescription;

}

// Getters

public int getPolicyId() {

return policyId;

}

public String getPolicyName() {

return policyName;

}

public String getRuleSetDescription() {

return ruleSetDescription;

}

//για τις μεθοδους - η λογικη θα προστεθεί στα επομενα βηματα

public double getPenaltyPercentage(int remainingHours) {

System.out.println("CancellationPolicy.getPenaltyPercentage() placeholder called.");

return 0.0; // προσωρινη επιστροφη

}

public double calculatePenalty(double originalCost, int remainingHours) {

System.out.println("CancellationPolicy.calculatePenalty() placeholder called.");

return 0.0; // προσωρινη επιστροφη

}}

**public class Payment** {

// γνωρισματα

private int paymentId; // μοναδικό αναγνωριστικό πληρωμης

private int bookingId; // ID κράτησης

private double amount; // Ποσό πληρωμής/επιστροφής

private String paymentType; // πληρωμή π.χ. Credit Card/Refund

private LocalDateTime transactionDate; // ημερομηνία/ώρα συναλλαγής

private String status; // κατάσταση συναλλαγης π.χ. Paid

// Constructor

public Payment(int paymentId, int bookingId, double amount, String paymentType) {

this.paymentId = paymentId;

this.bookingId = bookingId;

this.amount = amount;

this.paymentType = paymentType;

this.transactionDate = LocalDateTime.now();

this.status = "Pending"; // αρχική κατασταση

}

// Getters

public int getPaymentId() {

return paymentId;

}

public int getBookingId() {

return bookingId;

}

public double getAmount() {

return amount;

}

public String getPaymentType() {

return paymentType;

}

public LocalDateTime getTransactionDate() {

return transactionDate;

}

public String getStatus() {

return status;

}

// Setters

public void setStatus(String status) {

this.status = status;

}

//για τις μεθόδους - η λογική θα προστεθεί μετα

public boolean processPayment() {

System.out.println("Payment.processPayment() placeholder called.");

return false;

}

public boolean processRefund(double refundAmount) {

System.out.println("Payment.processRefund() placeholder called.");

return false;

}

}

// System.java

import java.util.HashMap;

import java.util.Map;

import java.time.LocalDateTime; // Για τη δημιουργία Booking

**public class System** {

// χρησιμοποιούνται ως αποθετήρια δεδομένων

private Map<Integer, User> users;

private Map<Integer, Booking> bookings;

private Map<Integer, Customer> customers;

private CancellationPolicy defaultCancellationPolicy; // Σύνδεση με την πολιτική

// μετρητές για μοναδικά IDs

private int nextCancellationId = 1;

private int nextPaymentId = 1;

// Constructor

public System() {

this.users = new HashMap<>();

this.bookings = new HashMap<>();

this.customers = new HashMap<>();

// αρχικοποίηση μιας default πολιτικής ακύρωσης χωρίς πλήρη λογική

this.defaultCancellationPolicy = new CancellationPolicy(1, "Standard", "Default rules for cancellation.");

// initializeDummyData(); θα ενεργοποιηθεί σε επομενο βημα για να αρχικοποιήσει δεδομένα

}

//η λογική θα προστεθεί σε επομενο βημα

public boolean cancelBooking(int userId, int bookingId) {

System.out.println("System.cancelBooking() placeholder called for UserID: " + userId + ", BookingID: " + bookingId);

return false; // Προσωρινή επιστροφή

}

public CancellationPolicy getCancellationPolicy() {

System.out.println("System.getCancellationPolicy() placeholder called.");

return defaultCancellationPolicy;

}

public double applyPenalty(Booking booking, CancellationPolicy policy, int remainingHours) {

System.out.println("System.applyPenalty() placeholder called.");

return 0.0;

}

public boolean processRefund(Booking booking, double refundAmount)

{

System.out.println("System.processRefund() placeholder called.");

return false; // Προσωρινή επιστροφή

}

public String displayCancellationConfirmation(Booking booking, Cancellation cancellation) {

System.out.println("System.displayCancellationConfirmation() placeholder called.");

return "Placeholder Confirmation Message"; // προσωρινα

}

// Βοηθητικές μέθοδοι για την προσθήκη αντικειμένων

public void addUser(User user) {

this.users.put(user.getUserId(), user);

}

public void addBooking(Booking booking) {

this.bookings.put(booking.getBookingId(), booking);

}

public void addCustomer(Customer customer) {

this.customers.put(customer.getCustomerId(), customer);

}

}