1. Examine an ATM System: Select an ATM system, thoroughly review

its system specifications, and identify and document the various software

defects present.

class ATM {

private int pin = 1234;

private double balance = 500;

private boolean auth = false;

public boolean login(int p) {

auth = (p == pin);

return auth;

}

public double getBalance() { return balance; }

public boolean withdraw(double amt) {

if (!auth) { System.out.println("Error: Invalid PIN."); return false; }

if (amt <= 0 || amt > balance) { System.out.println("Error: Invalid amount."); return false; }

balance -= amt;

System.out.println("Dispensed $" + amt);

System.out.println("Receipt Printed");

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

return true;

}

}

public class ATMTest {

public static void main(String[] args) {

ATM atm = new ATM();

System.out.println("Input: User selects withdrawal option, enters PIN=1234, requests $100");

if (atm.login(1234)) {

atm.withdraw(100);

} else {

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

}

}

}

1. Case Study on ARMS Portals: Analyzing Requirements and Developing Positive Test Scenarios

class ARMSPortal {

private String user = "testUser";

private String pass = "testPass";

public boolean login(String u, String p) {

return u.equals(user) && p.equals(pass);

}

public String dashboard() {

return "Welcome to Dashboard";

}

}

public class ARMSPortalTest {

public static void main(String[] args) {

ARMSPortal portal = new ARMSPortal();

System.out.println("Input: username=testUser, password=testPass");

if (portal.login("testUser", "testPass")) {

System.out.println("Output: " + portal.dashboard());

} else {

System.out.println("Output: Invalid Login");

}

}

}

1. Case Study on ARMS Portals: Analyzing Requirements and Developing Negative Test

Scenarios

class ARMSPortal {

private String user = "testUser";

private String pass = "testPass";

public boolean login(String u, String p) {

return u.equals(user) && p.equals(pass);

}

}

public class ARMSPortalNegativeTest {

public static void main(String[] args) {

ARMSPortal portal = new ARMSPortal();

System.out.println("Input: username=wrongUser, password=wrongPass");

if (portal.login("wrongUser", "wrongPass")) {

System.out.println("Output: Login Success (Error in handling!)");

} else {

System.out.println("Output: Invalid username or password");

}

}

}

1. Case Study on e-Commerce Applications: Analyzing Requirements and Creating Positive Test Scenarios

class ECommerceApp {

private String cart = "";

public void browse(String product) {

System.out.println("Product Selected: " + product);

cart = product;

}

public void checkout() {

if (!cart.isEmpty()) System.out.println("Proceeding to Checkout with " + cart);

}

public boolean payment(String card) {

return card.equals("validCard");

}

public void confirmOrder() {

System.out.println("Order Confirmed and Added to History");

}

}

public class ECommercePositiveTest {

public static void main(String[] args) {

ECommerceApp app = new ECommerceApp();

System.out.println("Input: User selects product, adds to cart, checkout, enters valid payment, confirms order");

app.browse("Laptop");

app.checkout();

if (app.payment("validCard")) {

System.out.println("Output: Payment Successful");

app.confirmOrder();

} else {

System.out.println("Output: Payment Failed");

}

}

}

1. Case Study on e-Commerce Applications: Analyzing Requirements and Creating Negative Test Scenarios

class ECommerceApp {

private String cart = "";

public void browse(String product) {

System.out.println("Product Selected: " + product);

cart = product;

}

public void checkout() {

if (cart.isEmpty()) {

System.out.println("Output: Please add items to your cart");

} else {

System.out.println("Proceeding to Checkout with " + cart);

}

}

public boolean payment(String card) {

if (!card.equals("validCard")) {

System.out.println("Output: Invalid payment details. Please try again.");

return false;

}

return true;

}

}

public class ECommerceNegativeTest {

public static void main(String[] args) {

ECommerceApp app = new ECommerceApp();

System.out.println("Input: User tries checkout without adding item to cart");

app.checkout();

System.out.println("\nInput: User adds product but enters invalid payment info");

app.browse("Mobile");

app.checkout();

if (!app.payment("wrongCard")) {

System.out.println("Transaction Failed");

}

}

}

1. Case Study on Food Delivery Applications Analyzing Requirements and Creating Positive Test Scenarios

class FoodDeliveryApp {

private String cart = "";

public void browseMenu(String item) {

System.out.println("Menu Item Selected: " + item);

cart = item;

}

public void checkout() {

if (!cart.isEmpty()) {

System.out.println("Proceeding to Checkout with " + cart);

}

}

public boolean payment(String card) {

return card.equals("validCard");

}

public void confirmOrder() {

System.out.println("Order Confirmed | Estimated Delivery: 30 minutes");

}

}

public class FoodDeliveryPositiveTest {

public static void main(String[] args) {

FoodDeliveryApp app = new FoodDeliveryApp();

System.out.println("Input: User browses menu, selects items, adds to cart, checkout, enters valid payment, confirms order");

app.browseMenu("Pizza");

app.checkout();

if (app.payment("validCard")) {

System.out.println("Output: Payment Successful");

app.confirmOrder();

} else {

System.out.println("Output: Payment Failed");

}

}

}

1. Case Study on Food Delivery Applications Analyzing Requirements and Creating Negative Test Scenarios

class FoodDeliveryApp {

private String cart = "";

public void checkout() {

if (cart.isEmpty()) System.out.println("Output: Your cart is empty. Please add items.");

}

public boolean payment(String card) {

if (!card.equals("validCard")) {

System.out.println("Output: Invalid payment method. Please check your card details.");

return false;

}

return true;

}

}

public class FoodDeliveryNegativeTest {

public static void main(String[] args) {

FoodDeliveryApp app = new FoodDeliveryApp();

System.out.println("Input: Checkout without selecting items");

app.checkout();

System.out.println("\nInput: User provides invalid card");

if (!app.payment("wrongCard")) System.out.println("Transaction Failed");

}

}

1. Case Study on Banking Applications: Analyzing Requirements and Developing Positive Test Scenarios

class BankingApp {

private String user = "admin", pass = "1234";

private double balance = 1000;

public boolean login(String u, String p) { return u.equals(user) && p.equals(pass); }

public boolean transfer(double amt) {

if (amt <= balance) {

balance -= amt;

System.out.println("Output: Transfer Successful | New Balance: " + balance);

return true;

}

return false;

}

}

public class BankingPositiveTest {

public static void main(String[] args) {

BankingApp app = new BankingApp();

System.out.println("Input: Valid login and transfer 500");

if (app.login("admin", "1234")) app.transfer(500);

else System.out.println("Output: Invalid Login");

}

}

1. Case Study on Banking Applications: Analyzing Requirements and Developing Negative Test Scenarios

class BankingApp {

private String user = "admin", pass = "1234";

private double balance = 500;

public boolean login(String u, String p) { return u.equals(user) && p.equals(pass); }

public boolean transfer(double amt) {

if (amt > balance) {

System.out.println("Output: Insufficient funds. Please check your balance.");

return false;

}

return true;

}

}

public class BankingNegativeTest {

public static void main(String[] args) {

BankingApp app = new BankingApp();

System.out.println("Input: Invalid login attempt");

if (!app.login("admin", "wrong")) System.out.println("Output: Invalid username or password");

System.out.println("\nInput: Transfer exceeding balance");

if (!app.transfer(1000)) System.out.println("Transaction Failed");

}

}

1. Test Plan Development for Library Management System

class LibrarySystem {

private String user = "admin", pass = "admin123";

public boolean login(String u, String p) { return u.equals(user) && p.equals(pass); }

}

public class LMSTest {

public static void main(String[] args) {

LibrarySystem lms = new LibrarySystem();

System.out.println("Input 1: username=admin, password=admin123");

if (lms.login("admin", "admin123")) System.out.println("Output: Login Successful");

System.out.println("\nInput 2: username=admin, password=wrongpassword");

if (!lms.login("admin", "wrongpassword")) System.out.println("Output: Invalid username or password");

}

}