import javax.swing.\*;import java.awt.\*;import java.util.ArrayList;import java.util.HashMap;public class OnlineBookStore { private static final HashMap<String, ArrayList<Book>> genres = new HashMap<>(); private static final ArrayList<Book> cart = new ArrayList<>(); private static final ArrayList<Book> purchasedBooks = new ArrayList<>(); private static final Color[] bookColors = {Color.PINK, Color.CYAN, Color.ORANGE, Color.YELLOW, Color.MAGENTA}; public static void main(String[] args) { initializeBooks(); JFrame frame = new JFrame("Online Book Store"); frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); frame.setSize(1000, 600); frame.setLayout(new BorderLayout()); // Sidebar JPanel sidebar = new JPanel(); sidebar.setBackground(new Color(200, 220, 240)); sidebar.setPreferredSize(new Dimension(200, frame.getHeight())); sidebar.setLayout(new BoxLayout(sidebar, BoxLayout.Y\_AXIS)); JLabel title = new JLabel("Book Store"); title.setFont(new Font("Arial", Font.BOLD, 20)); title.setAlignmentX(Component.CENTER\_ALIGNMENT); sidebar.add(Box.createRigidArea(new Dimension(0, 20))); sidebar.add(title); sidebar.add(Box.createRigidArea(new Dimension(0, 20))); JButton exploreButton = new JButton("Explore"); JButton categoriesButton = new JButton("Categories"); JButton cartButton = new JButton("Cart"); JButton purchaseButton = new JButton("Purchase"); addSidebarButton(exploreButton, sidebar); addSidebarButton(categoriesButton, sidebar); addSidebarButton(cartButton, sidebar); addSidebarButton(purchaseButton, sidebar); frame.add(sidebar, BorderLayout.WEST); // Main content JPanel mainContent = new JPanel(); mainContent.setBackground(new Color(240, 240, 240)); mainContent.setLayout(new BorderLayout()); frame.add(mainContent, BorderLayout.CENTER); // Header with search JPanel header = new JPanel(new BorderLayout()); header.setBackground(new Color(200, 220, 240)); header.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10)); JTextField searchBox = new JTextField(" Search for books..."); JButton searchButton = new JButton("Search"); header.add(searchBox, BorderLayout.CENTER); header.add(searchButton, BorderLayout.EAST); mainContent.add(header, BorderLayout.NORTH); // Book display panel JPanel bookDisplay = new JPanel(new GridLayout(0, 3, 10, 10)); bookDisplay.setBackground(new Color(240, 240, 240)); bookDisplay.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20)); mainContent.add(bookDisplay, BorderLayout.CENTER); // Event Listeners exploreButton.addActionListener(e -> displayExplorerBooks(bookDisplay)); categoriesButton.addActionListener(e -> displayGenres(bookDisplay)); cartButton.addActionListener(e -> displayCart(bookDisplay)); purchaseButton.addActionListener(e -> purchaseBooks(bookDisplay)); searchButton.addActionListener(e -> { String query = searchBox.getText().trim(); searchBooks(query, bookDisplay); }); // Show the frame frame.setVisible(true); } private static void initializeBooks() { genres.put("Fiction", new ArrayList<>()); genres.put("Non-Fiction", new ArrayList<>()); genres.put("Science", new ArrayList<>()); genres.put("Technology", new ArrayList<>()); genres.put("History", new ArrayList<>()); // Assign unique names for each book String[] fictionTitles = {"The Hobbit", "Harry Potter", "The Alchemist", "Lord of the Rings", "Percy Jackson", "Dune", "The Shining", "1984", "Animal Farm", "Dracula"}; String[] nonFictionTitles = {"Sapiens", "Educated", "Becoming", "The Wright Brothers", "Into the Wild", "The Immortal Life of Henrietta Lacks", "Outliers", "The Glass Castle", "Unbroken", "Steve Jobs"}; String[] scienceTitles = {"Brief History of Time", "Cosmos", "Astrophysics for People in a Hurry", "The Gene", "Sapiens", "The Selfish Gene", "Origin of Species", "What If?", "The Elegant Universe", "The Universe in a Nutshell"}; String[] techTitles = {"Clean Code", "The Pragmatic Programmer", "Artificial Intelligence", "Introduction to Algorithms", "The Mythical Man-Month", "Design Patterns", "Code Complete", "You Don't Know JS", "Deep Learning", "Data Structures and Algorithms"}; String[] historyTitles = {"The Silk Roads", "Guns, Germs, and Steel", "The History of the World", "The Roman Empire", "The Diary of Anne Frank", "A People's History of the United States", "The Crusades", "The Cold War", "World War II", "The Vietnam War"}; for (int i = 0; i < 10; i++) { genres.get("Fiction").add(new Book(fictionTitles[i], 150 + i \* 10)); genres.get("Non-Fiction").add(new Book(nonFictionTitles[i], 200 + i \* 10)); genres.get("Science").add(new Book(scienceTitles[i], 250 + i \* 15)); genres.get("Technology").add(new Book(techTitles[i], 300 + i \* 20)); genres.get("History").add(new Book(historyTitles[i], 350 + i \* 25)); } } private static void addSidebarButton(JButton button, JPanel sidebar) { button.setFocusPainted(false); button.setAlignmentX(Component.CENTER\_ALIGNMENT); button.setBackground(Color.WHITE); button.setForeground(Color.DARK\_GRAY); button.setBorder(BorderFactory.createEmptyBorder(20, 10, 20, 10)); sidebar.add(button); sidebar.add(Box.createRigidArea(new Dimension(0, 10))); } private static void displayExplorerBooks(JPanel bookDisplay) { bookDisplay.removeAll(); ArrayList<Book> famousBooks = new ArrayList<>(); famousBooks.add(new Book("To Kill a Mockingbird", 300)); famousBooks.add(new Book("1984", 250)); famousBooks.add(new Book("The Great Gatsby", 200)); famousBooks.add(new Book("The Catcher in the Rye", 250)); famousBooks.add(new Book("Pride and Prejudice", 350)); for (Book book : famousBooks) { displayBookCard(book, bookDisplay); } bookDisplay.revalidate(); bookDisplay.repaint(); } private static void displayGenres(JPanel bookDisplay) { bookDisplay.removeAll(); for (String genre : genres.keySet()) { JButton genreButton = new JButton(genre); genreButton.addActionListener(e -> displayBooksByGenre(genre, bookDisplay)); bookDisplay.add(genreButton); } bookDisplay.revalidate(); bookDisplay.repaint(); } private static void displayBooksByGenre(String genre, JPanel bookDisplay) { bookDisplay.removeAll(); ArrayList<Book> books = genres.get(genre); books.forEach(book -> displayBookCard(book, bookDisplay)); bookDisplay.revalidate(); bookDisplay.repaint(); } private static void displayBookCard(Book book, JPanel bookDisplay) { JPanel bookCard = new JPanel(); bookCard.setLayout(new BorderLayout()); bookCard.setBorder(BorderFactory.createLineBorder(Color.DARK\_GRAY, 1)); bookCard.setBackground(bookColors[(int) (Math.random() \* bookColors.length)]); JLabel bookTitle = new JLabel(book.getName()); bookTitle.setHorizontalAlignment(SwingConstants.CENTER); JLabel bookPrice = new JLabel("Price: $" + book.getPrice()); bookPrice.setHorizontalAlignment(SwingConstants.CENTER); JButton addToCartButton = new JButton("Add to Cart"); addToCartButton.addActionListener(e -> cart.add(book)); bookCard.add(bookTitle, BorderLayout.NORTH); bookCard.add(bookPrice, BorderLayout.CENTER); bookCard.add(addToCartButton, BorderLayout.SOUTH); bookDisplay.add(bookCard); } private static void displayCart(JPanel bookDisplay) { bookDisplay.removeAll(); if (cart.isEmpty()) { JLabel emptyCartLabel = new JLabel("Your cart is empty."); emptyCartLabel.setFont(new Font("Arial", Font.BOLD, 16)); emptyCartLabel.setHorizontalAlignment(SwingConstants.CENTER); bookDisplay.add(emptyCartLabel); } else { JPanel cartPanel = new JPanel(); cartPanel.setLayout(new BoxLayout(cartPanel, BoxLayout.Y\_AXIS)); cartPanel.setAlignmentX(Component.CENTER\_ALIGNMENT); cart.forEach(book -> { JLabel cartItemLabel = new JLabel(book.getName() + " - $" + book.getPrice()); cartItemLabel.setAlignmentX(Component.CENTER\_ALIGNMENT); cartPanel.add(cartItemLabel); }); bookDisplay.add(cartPanel); } bookDisplay.revalidate(); bookDisplay.repaint(); } private static void purchaseBooks(JPanel bookDisplay) { bookDisplay.removeAll(); if (cart.isEmpty()) { JLabel emptyCartLabel = new JLabel("Your cart is empty."); bookDisplay.add(emptyCartLabel); } else { purchasedBooks.addAll(cart); int totalPrice = cart.stream().mapToInt(Book::getPrice).sum(); JPanel purchasePanel = new JPanel(); purchasePanel.setLayout(new BoxLayout(purchasePanel, BoxLayout.Y\_AXIS)); JLabel successLabel = new JLabel("Books purchased successfully!"); successLabel.setFont(new Font("Arial", Font.BOLD, 16)); successLabel.setAlignmentX(Component.CENTER\_ALIGNMENT); purchasePanel.add(successLabel); JLabel totalPriceLabel = new JLabel("Total Price: $" + totalPrice); totalPriceLabel.setFont(new Font("Arial", Font.PLAIN, 14)); totalPriceLabel.setAlignmentX(Component.CENTER\_ALIGNMENT); purchasePanel.add(totalPriceLabel); JLabel purchasedBooksLabel = new JLabel("Purchased Books:"); purchasedBooksLabel.setFont(new Font("Arial", Font.BOLD, 14)); purchasedBooksLabel.setAlignmentX(Component.CENTER\_ALIGNMENT); purchasePanel.add(purchasedBooksLabel); purchasedBooks.forEach(book -> { JLabel bookLabel = new JLabel("- " + book.getName() + " ($" + book.getPrice() + ")"); bookLabel.setAlignmentX(Component.CENTER\_ALIGNMENT); purchasePanel.add(bookLabel); }); bookDisplay.add(purchasePanel); cart.clear(); } bookDisplay.revalidate(); bookDisplay.repaint(); } private static void searchBooks(String query, JPanel bookDisplay) { bookDisplay.removeAll(); boolean bookFound = false; for (ArrayList<Book> books : genres.values()) { for (Book book : books) { if (book.getName().toLowerCase().contains(query.toLowerCase())) { displayBookCard(book, bookDisplay); bookFound = true; } } } if (!bookFound) { JLabel notFoundLabel = new JLabel("Book not found"); notFoundLabel.setFont(new Font("Arial", Font.BOLD, 16)); notFoundLabel.setHorizontalAlignment(SwingConstants.CENTER); bookDisplay.add(notFoundLabel); } bookDisplay.revalidate(); bookDisplay.repaint(); }}class Book { private final String name; private final int price; public Book(String name, int price) { this.name = name; this.price = price; } public String getName() { return name; } public int getPrice() { return price; }}