```java
package App;

public class Parts {
    protected int id;
    protected String name;
    protected String type;
    protected String brand;
    protected int price;
    protected int volume;

    // Constructor
    public Parts(int id, String name, String type, String brand, int price, int volume) {
        this.id = id;
        this.name = name;
        this.type = type;
        this.brand = brand;
        this.price = price;
        this.volume = volume;
    }

    // Factory method to create Parts object from CSV line
    public static Parts createFromCSVLine(String line) {
        String[] data = line.split(",");
        int id = Integer.parseInt(data[0].trim());
        String name = data[1].trim();
        String type = data[2].trim();
        String brand = data[3].trim();
        int price = Integer.parseInt(data[4].trim());
        int volume = Integer.parseInt(data[5].trim());

        if (type.equalsIgnoreCase("Engine")) {
            // Parse additional data for Engine
            int horsepower = Integer.parseInt(data[6].trim());
            return new Engine(id, name, type, brand, price, volume, horsepower);
        }
        else if(type.equalsIgnoreCase("Wheels")){
            int diameter = Integer.parseInt(data[6].trim());
            return new Wheels(id, name, type, brand, price, volume, diameter);
        }
        else if(type.equalsIgnoreCase("Turbo")){
            String boost = (data[6].trim());
            return new Turbo(id, name, type, brand, price, volume, boost);
        }
        else if(type.equalsIgnoreCase("Ecu")){
            return new Ecu(id, name, type, brand, price, volume);
        }
        else if(type.equalsIgnoreCase("Rear wing")){
            String material = (data[6].trim());
            return new RearWing(id, name, type, brand, price, volume, material);
```

```
        }
        else if(type.equalsIgnoreCase("Aero Kit")){
            String color = (data[6].trim());
            return new AeroKit(id, name, type, brand, price, volume,
color);
        }
        else {
            // For other types, create a regular Parts object
            return new Parts(id, name, type, brand, price, volume);
        }
    }
}

public class PartPrinter {

    public static void printPartInfo(Parts part) {
        System.out.println("        +------------------------------------>");
        System.out.println("        |      " +
part.getClass().getSimpleName() );
        System.out.println("        +------------------------------------>");
        System.out.println("        +------------------------------------>");
        System.out.println("        | - ID: " + part.id);
        System.out.println("        | - Name: " + part.name);
        System.out.println("        | - Type: " + part.type);
        System.out.println("        | - Brand: " + part.brand);
        System.out.println("        | - Price: " + part.price+"$");

        if (part instanceof Engine) {
            System.out.println("        | - Horse Power: " + ((Engine)
part).getHorsepower()+" BHP");
        }
        else if(part instanceof Wheels){
            System.out.println("        | - Wheel Size: " + ((Wheels)
part).getDiameter()+" inches");
        }
        else if(part instanceof Turbo){
            System.out.println("        | - Boost max: " + ((Turbo)
part).getBoost()+" psi");
        }
        else if(part instanceof RearWing){
            System.out.println("        | - Wing material: " + ((RearWing)
part).getMaterial());
        }
        else if(part instanceof AeroKit){
            System.out.println("        | - Color available: " +
((AeroKit) part).getColor());
        }
        System.out.println("        | - Available parts: " + part.volume);
        System.out.println("        +------------------------------------>");
    }
}

package App;

public class CallBack {
```

```
    public CallBack() {
        try {
            System.out.println("\tInstert 1 to find the Nearest dealer\n\t\t2 to go back to menu");
            System.out.print("\tInsert here: ");
            int n;
            while (true) {
                try {
                    n = AppScanner.nextInt();
                    break;
                } catch (Exception e) {
                    System.out.println("Invalid input. Please enter a number.");
                }
            }

            if (n == 1) {
                new LocationFinder();
            } else if (n == 2) {
                new Menu();
            } else {
                while (n ≠ 1 && n ≠ 2) {
                    System.out.println("\tInvalid range. \n\tPlease insert either 1 or 2.");
                    System.out.print("\tSelect range: ");

                    while (true) {
                        try {
                            n = AppScanner.nextInt();
                            break;
                        } catch (Exception e) {
                            System.out.println("Invalid input. Please enter a number.");
                        }
                    }
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

//java file for writing on CSV logic
package App;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;
```

```
149
150 public class CSVWriter {
151     private static final String CSV_FILE_PATH = "Database/PartsData.csv";
152
153     // Method to update the volume of a part in the CSV file
154     public static void updateVolume(int partId, int newVolume) {
155         try {
156             List<String> lines =
    Files.readAllLines(Paths.get(CSV_FILE_PATH));
157
158             // Adjust the part index based on the starting ID (1000 in
    your case)
159             int partIndex = partId - 999;
160
161             if (partIndex >= 0 && partIndex < lines.size()) {
162                 String[] parts = lines.get(partIndex).split(",");
163                 parts[5] = String.valueOf(newVolume);
164                 lines.set(partIndex, String.join(",", parts));
165
166                 Files.write(Paths.get(CSV_FILE_PATH), lines);
167                 System.out.println("Volume updated successfully!");
168             } else {
169                 System.out.println("Invalid part ID.");
170             }
171         } catch (IOException e) {
172             e.printStackTrace();
173         }
174     }
175
176     public static void addNewPart() {
177         // Read existing lines to find the highest index
178         int highestIndex = findHighestIndex();
179         int newIndex = highestIndex + 1;
180
181         System.out.println("Adding a new part with index: " + newIndex);
182
183         // Gather information from the user
184         System.out.print("Enter part name: ");
185         String name = AppScanner.nextLine();
186
187         System.out.print("Enter part type: ");
188         String type = AppScanner.nextLine();
189
190         System.out.print("Enter part brand: ");
191         String brand = AppScanner.nextLine();
192
193         System.out.print("Enter part price: ");
194         int price = AppScanner.nextInt();
195
196         System.out.print("Enter part volume: ");
197         int volume = AppScanner.nextInt();
198
199         String additionalInfo = "";
200
201         // Based on part type, ask for additional information
```

```
202         switch (type.toLowerCase()) {
203             case "engine":
204                 System.out.print("Enter horsepower: ");
205                 additionalInfo = AppScanner.nextLine();
206                 break;
207             case "wheels":
208                 System.out.print("Enter diameter: ");
209                 additionalInfo = AppScanner.nextLine();
210                 break;
211             case "turbo":
212                 System.out.print("Enter boost: ");
213                 additionalInfo = AppScanner.nextLine();
214                 break;
215             case "ecu":
216                 // No additional information for ECU
217                 break;
218             case "rear wing":
219                 System.out.print("Enter material: ");
220                 additionalInfo = AppScanner.nextLine();
221                 break;
222             case "aero kit":
223                 System.out.print("Enter color: ");
224                 additionalInfo = AppScanner.nextLine();
225                 break;
226             default:
227                 System.out.println("Invalid part type.");
228                 return;
229         }
230 
231         // Construct the new part line
232         String newPartLine = newIndex + "," + name + "," + type + "," +
    brand + "," + price + "," + volume + "," + additionalInfo;
233 
234         // Append the new line to the CSV file
235         try (PrintWriter writer = new PrintWriter(new BufferedWriter(new
    FileWriter(CSV_FILE_PATH, true)))) {
236             writer.println(newPartLine);
237             System.out.println("New part added successfully.");
238 
239         } catch (IOException e) {
240             System.out.println("Error writing to the CSV file.");
241             e.printStackTrace();
242         }
243     }
244 
245     private static int findHighestIndex() {
246         int highestIndex = 0;
247 
248         try (BufferedReader br = new BufferedReader(new
    FileReader(CSV_FILE_PATH))) {
249             String line;
250             boolean isFirstLine = true;
251 
252             while ((line = br.readLine()) != null) {
253                 if (isFirstLine) {
```

```
                    isFirstLine = false;
                    continue;
                }

                String[] parts = line.split(",");
                int currentId = Integer.parseInt(parts[0]);
                highestIndex = Math.max(highestIndex, currentId);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

        return highestIndex;
    }
}

package App;

public class LoginVerification {

    public LoginVerification() {
        boolean loginSuccessful = false;

        do {
            System.out.print("\tPlease input User Id: ");
            String id = AppScanner.nextLine();
            System.out.print("\tPlease input Password: ");
            String pass = AppScanner.nextLine();

            // Default login admin admin
            if (id.equals("admin") && pass.equals("admin")) {
                // Login successful
                loginSuccessful = true;
                new AdminPanel();
            } else {
                System.out.println("\tPress R to retry or M to go back to
menu :");
                String c = AppScanner.nextLine();

                if (c.equalsIgnoreCase("M")) {
                    new Menu();
                    break;
                }
            }
        } while (!loginSuccessful);
    }
}


```