CAMERARENTAL APPLICATION:

```
package sample_project;

	import java.util.ArrayList;
	import java.util.List;
	import java.util.Scanner;



public class Camerarentall {
		private static double walletBalance = 0.0;
		private static List<Camera> cameras = new ArrayList<>();

		public static void main(String[] args) {
			Scanner scanner = new Scanner(System.in);

			while (true) {
				System.out.println("+--------------------------------+");
				System.out.println("|      WELCOME TO RENTAL APP      |");
				System.out.println("+--------------------------------+");
				System.out.println("Hello Chief!! Let's Login to continue,");

				String username;

				do {
```

```
                System.out.print("Enter username: ");
                username = scanner.nextLine().trim();

                if (username.isEmpty()) {
                        System.out.println("Chief,Username cannot be
empty. Please try again.\n");
                }
            } while (username.isEmpty());

            String password;

            do {
                System.out.print("Enter password: ");
                password = scanner.nextLine().trim();

                if (password.isEmpty()) {
                        System.out.println("Chief,Password cannot be
empty. Please try again.\n");
                }
            } while (password.isEmpty());

            // Check login credentials
            if (username.equals("admin") && password.equals("password")) {
                System.out.println("\nWelcome Back Chief! Login
successful,");
                break;
```

```
                } else {
                        System.out.println("Chief,Invalid username or password.
Please try again.\n");
                }
            }

            while (true) {
                System.out.println();
                System.out.println("Options:");
                System.out.println("1. Add camera");
                System.out.println("2. Remove camera");
                System.out.println("3. Rent a camera");
                System.out.println("4. View all cameras");
                System.out.println("5. My wallet or Add money");
                System.out.println("6. Search cameras by brand or model");
                System.out.println("7. Sort cameras");
                System.out.println("8. Exit");

                int option = 0;
                boolean isValidOption = false;

                do {
                    System.out.print("Select an option: ");
                    try {
                        option = Integer.parseInt(scanner.nextLine());
```

```
                    if (option < 1 || option > 8) {
                        throw new IllegalArgumentException("\nChief, Invalid option. Please select a valid option.");
                    }

                    isValidOption = true;
                } catch (NumberFormatException e) {
                    System.out.println("\nChief, Option Cannot be empty (or)Not to be in String");
                } catch (IllegalArgumentException e) {
                    System.out.println(e.getMessage());
                }
            } while (!isValidOption);

                switch (option) {
                case 1:
                        addCamera(scanner);
                        break;
                case 2:
                        removeCamera(scanner);
                        break;
                case 3:
                        rentCamera(scanner);
                        break;
                case 4:
```

```
                    viewAllCameras();
                    break;
                case 5:
                    myWallet(scanner);
                    break;
                case 6:
                    searchCameras(scanner);
                    break;
                case 7:
                    sortCameras();
                    break;
                case 8:
                    System.out.println("Chief, Exirted Successfully");
                    System.exit(0);
                default:
                    System.out.println("Chief,Invalid option. Please try
again.");
                }
            }
        }

        private static void addCamera(Scanner scanner) {
            String brand;
            do {
                System.out.print("\nEnter brand: ");
```

```
            brand = scanner.nextLine().trim();

            if (brand.isEmpty()) {
                System.out.println("Chief, Brand cannot be empty.");
            }
        } while (brand.isEmpty());

        String model;
        do {
            System.out.print("Enter model: ");
            model = scanner.nextLine().trim();

            if (model.isEmpty()) {
                System.out.println("Chief,Model cannot be empty.\n");
            }
        } while (model.isEmpty());

        double rentalAmount;
        do {
            System.out.print("Enter rental amount: ");
            String rentalAmountInput = scanner.nextLine().trim();

            try {
                rentalAmount = Double.parseDouble(rentalAmountInput);
                if (rentalAmount <= 0) {
```

```
                                        throw new IllegalArgumentException("Chief,
Rental amount must be greater than 0.");
                                    }
                                } catch (NumberFormatException e) {
                                    System.out.println("\nChief, Invalid rental amount. Please
enter a valid number.");
                                    rentalAmount = -1;
                                } catch (IllegalArgumentException e) {
                                    System.out.println(e.getMessage());
                                    rentalAmount = -1;
                                }
                            } while (rentalAmount <= 0);

                            Camera camera = new Camera(brand, model, rentalAmount);
                            cameras.add(camera);

                            System.out.println("\nChief, Camera was added successfully.");
                            System.out.println("Status: Available");
                        }

                        private static void removeCamera(Scanner scanner) {
                            if (cameras.isEmpty()) {
                                System.out.println("\nChief, There are no cameras available to remove.");
                                return;
                            }
```

```
                System.out.println("\nAvailable Cameras to Remove:");
                System.out.println("+-------------------------------------------------------------------
----+");
                System.out.println("| Index | Brand          | Model          | Rental Amount   |
Status    |");
                System.out.println("+-------+----------------+----------------+----------------+-----
------+");
                for (int i = 0; i < cameras.size(); i++) {
                    Camera camera = cameras.get(i);
                    System.out.printf("| %-5d | %-15s | %-15s | $%-12.2f  | %-9s |\n", i,
camera.getBrand(), camera.getModel(),
                            camera.getRentalAmount(), camera.getStatus());
                }
                System.out.println("+-------+----------------+----------------+----------------+-----
------+");


                int index;
                while (true) {
                    System.out.print("Chief, Enter the index of the camera to remove: ");
                    String input = scanner.nextLine();


                    if (input.isEmpty()) {
                        System.out.println("\nChief, Index value cannot be empty. Please try
again.");
                        continue;
                    }
```

```
            try {
                index = Integer.parseInt(input);
                break;
            } catch (NumberFormatException e) {
                System.out.println("Chief, Invalid index. Please enter a valid number.");
            }
        }

        if (index >= 0 && index < cameras.size()) {
            cameras.remove(index);
            System.out.println("Chief, Camera was removed successfully.");
        } else {
            System.out.println("Chief, Invalid index. Please try again.");
        }
    }


    private static void rentCamera(Scanner scanner) {
        if (cameras.isEmpty()) {
            System.out.println("\nChief, No cameras available to rent.");
            return;
        }

        System.out.println("\nAvailable Cameras:");
```

```
                System.out.println("+-------------------------------------------------------------------------
----+");
                System.out.println("| Index | Brand          | Model          | Rental Amount   |
Status    |");
                System.out.println("+-------+----------------+----------------+----------------+-----
------+");
                for (int i = 0; i < cameras.size(); i++) {
                    Camera camera = cameras.get(i);
                    System.out.printf("| %-5d | %-15s | %-15s | $%-12.2f   | %-9s |\n", i,
camera.getBrand(), camera.getModel(),
                            camera.getRentalAmount(), camera.getStatus());
                }
                System.out.println("+-------+----------------+----------------+----------------+-----
------+");

                System.out.println("-- Available balance: $" + walletBalance + " --");

                int index = -1;
                boolean isValidIndex = false;

                do {
                    System.out.print("\nChief, enter the index of the camera to rent: ");
                    try {
                        index = Integer.parseInt(scanner.nextLine());

                        if (index >= 0 && index < cameras.size()) {
                            isValidIndex = true;
```

```
                    } else {
                        System.out.println("Chief, Enter a valid index number.");
                    }
                } catch (NumberFormatException e) {
                    System.out.println("Chief, Enter a valid index number.");
                }
            } while (!isValidIndex);

            Camera camera = cameras.get(index);
            if (camera.getStatus().equals("Available")) {
                double rentalAmount = camera.getRentalAmount();
                if (walletBalance >= rentalAmount) {
                    walletBalance -= rentalAmount;
                    camera.setStatus("Rented");

                    System.out.println("+-------------------------------------------------------+");
                    System.out.println("|                Rental Status                          |");
                    System.out.println("+-------------------------------------------------------+");
                    System.out.println("| Brand      | Model          | Rental Amount   | Status |");
                    System.out.println("+------------+----------------+-----------------+----------+");
                    System.out.printf("| %-10s | %-15s | $%-15.2f| Rented  |\n", camera.getBrand(), camera.getModel(),
                            rentalAmount);
                    System.out.println("+------------+----------------+-----------------+----------+");
```

```
                    System.out.println("Status: Rented");
                    System.out.println("Chief, Camera rented successfully.");
                    System.out.println("-- Remaining balance: $" + walletBalance + " --");
                } else {
                    System.out.println("Chief, Transaction Failed!! Insufficient balance in your wallet. Add Money");
                }
            } else {
                System.out.println("Chief, Camera is not available for rent.");
            }
        }

        private static void viewAllCameras() {
            if (cameras.isEmpty()) {
                System.out.println("Chief, No cameras available.");
                return;
            }

            System.out.println("\nAll Cameras:");
            System.out.println("+-------------------------------------------------------------------+");
            System.out.println("| Index | Brand          | Model          | Rental Amount | Status    |");
            System.out.println("+-------+----------------+----------------+---------------+-----------+");
            for (int i = 0; i < cameras.size(); i++) {
```

```
                    Camera camera = cameras.get(i);
                    System.out.printf("| %-5d | %-15s | %-15s | $%-13.2f  | %-9s |\n", i,
camera.getBrand(), camera.getModel(),
                            camera.getRentalAmount(), camera.getStatus());
                }
                System.out.println("+-------+-----------------+-----------------+-----------------
+-----------+");
            }

            private static void myWallet(Scanner scanner) {
                while (true) {
                    System.out.println("\nCurrent wallet balance: $" + walletBalance);
                    System.out.print("Chief, enter the amount to deposit: ");
                    String input = scanner.nextLine();

                    if (input.isEmpty()) {
                        System.out.println("Chief, wallet amount cannot be empty.");
                        continue;
                    }

                    try {
                        double amount = Double.parseDouble(input);
                        if (amount < 0) {
                            throw new IllegalArgumentException("Chief, amount cannot be
negative.");
                        }
```

```
                walletBalance += amount;
                System.out.println("\nChief, Deposit successful. Current wallet balance: $" + walletBalance);
                break;
            } catch (NumberFormatException e) {
                System.out.println("Chief, enter a valid amount.");
            } catch (IllegalArgumentException e) {
                System.out.println(e.getMessage());
            }
        }
    }

    private static void searchCameras(Scanner scanner) {
        try {
            if (cameras.isEmpty()) {
                throw new IllegalStateException("\nChief, No cameras available.");
            }

            while (true) {
                System.out.print("\nChief, Enter brand or model to search (or type 'back' to go options): ");
                String keyword = scanner.nextLine();

                if (keyword.equalsIgnoreCase("back")) {
                    break; // Go back to the options page
```

```
                }

                if (keyword.isEmpty()) {
                    System.out.println("Chief, Keyword cannot be empty.");
                    continue;
                }

                List<Camera> searchResults = new ArrayList<>();
                for (Camera camera : cameras) {
                    if (camera.getBrand().equalsIgnoreCase(keyword) ||
camera.getModel().equalsIgnoreCase(keyword)) {
                        searchResults.add(camera);
                    }
                }

                if (searchResults.isEmpty()) {
                    System.out.println("Chief, No cameras found matching the search.");
                } else {
                    System.out.println("Search Results:");
                    System.out.println("+-------------------------------------------------------------
-------------+");
                    System.out.println("| Index | Brand          | Model          | Rental
Amount   | Status    |");
                    for (int i = 0; i < searchResults.size(); i++) {
                        Camera camera = searchResults.get(i);
                        System.out.printf("| %-5d | %-15s | %-15s | $%-12.2f   | %-9s |\n", i,
camera.getBrand(),
```

```
                                camera.getModel(), camera.getRentalAmount(),
camera.getStatus());
                        }
                        System.out.println("+-------+----------------+----------------+-------------
---+-----------+");
                    }
                }
            } catch (IllegalStateException e) {
                System.out.println(e.getMessage());
            }
        }



        private static void sortCameras() {
            try {
                if (cameras.isEmpty()) {
                    throw new IllegalStateException("\nChief, No cameras available right
now.");
                }

                cameras.sort((c1, c2) ->
c1.getBrand().compareToIgnoreCase(c2.getBrand()));

                System.out.println("Cameras sorted by brand:");
```

```
                    System.out.println("+-------------------------------------------------------------------
-------+");
                    System.out.println("| Index | Brand          | Model          | Rental Amount   |
Status    |");
                    System.out.println("+-------+----------------+----------------+----------------+--
---------+");
                    for (int i = 0; i < cameras.size(); i++) {
                        Camera camera = cameras.get(i);
                        System.out.printf("| %-5d | %-15s | %-15s | $%-12.2f   | %-9s |\n", i,
camera.getBrand(), camera.getModel(),
                                camera.getRentalAmount(), camera.getStatus());
                    }
                    System.out.println("+-------+----------------+----------------+----------------+--
---------+");
                } catch (IllegalStateException e) {
                    System.out.println(e.getMessage());
                }

                // Take the user back to the options page
                showOptions();
            }

            private static void showOptions() {
                    // TODO Auto-generated method stub

            }
        }
```

```
class Camera {
	private String brand;
	private String model;
	private double rentalAmount;
	private String status;

	public Camera(String brand, String model, double rentalAmount) {
		this.brand = brand;
		this.model = model;
		this.rentalAmount = rentalAmount;
		this.status = "Available";
	}

	public String getBrand() {
		return brand;
	}

	public String getModel() {
		return model;
	}

	public double getRentalAmount() {
		return rentalAmount;
	}
```

```
	public String getStatus() {
		return status;
	}

	public void setStatus(String status) {
		this.status = status;
	}
}
```