Project 4: Customer Loyalty Management System

WM9A4DDP Class test

Author: Chuanxin Zhai 5546118

1. Documentation

1.1 The structure of the database:

The Customer Loyalty Management System is designed to enhance customer engagement and retention in an e-commerce environment. By tracking customer purchases, recording points, offering rewards, and generating loyalty reports, the system aims to create a rewarding shopping experience for loyal customers.

The customer loyalty management system uses a dictionary structure to store customer information, and each customer record includes the following fields:

name: customer name, serve as the key of the dictionary.
contact: customer contact details
points: customer loyalty points
redeemed: the points that customers have done redemption

sample:

```
customers = {
    "Jordan": {
        "contact": "Jordan@gmail.com",
        "points": 500,
        "redeemed": 1000
    }
}
```

1.2 key functions and realization

- Adding Customers: The add_customer function adds new customers to the database. It accepts the customer's name and contact information as parameters and creates a new record in the database.
- Recording Points: The record_points function records points for each transaction. It updates the customer's point balance by adding points based on the value of the purchase transaction.
- Displaying Points: The display_total_points function displays the total points for each customer. This allows both customers and administrators to know the point balance.
- Redeeming Rewards: The offer_rewards function allows customers to redeem rewards

using their accumulated points. It checks if the customer has enough points for the redemption and updates the point balance and redemption history accordingly.

- Loyalty Reports: The loyalty_report function generates reports on customer loyalty, including each customer's total points and redemption history.

1.3 Key Decisions in the Development Process

- Choice of Data Structure: The dictionary was chosen as the primary data structure as it provides quick key-value pair access, suitable for storing and retrieving customer information.

  Function Modularity: Each functionality of the system is designed as a separate function, enhancing code reusability and maintainability. For example, functions for adding customers, recording points, and redeeming rewards each have their own implementation.
- User Interface Design: An easy-to-use text-based user interface was implemented, allowing users to interact with the system through simple menu selections. This design makes the system accessible even to users unfamiliar with technology.
- Error Handling: Error handling logic is incorporated into the functionality implementation to ensure the system gracefully handles invalid inputs and exceptional situations. For example, attempting to record points for a non-existent customer prompts an error message.
- Status and Reporting: The system includes functionality to display loyalty status and generate loyalty reports, helping businesses and customers understand point usage and customer loyalty levels.

2. Test

2.1 normal cases

This is the interface:

```
1. add customers
2. record loyalty points
3. display loyalty points
4. redemption loyalty points for discount
5. display loyalty status
6. reset loyalty points
7. check loyalty reports
8. Exit
choose operation:
```

Input “1” add the customer’s information

```
choose operation: 1
input customer name: chesson
input customer contact details: chesson@gmail.com
```

Input “2” add the loyalty points for the customer

```
choose operation: 2
input customer name: chesson
input loyalty points: 1500
chessonloyalty points earned by this transaction: 1500
```

Input “3” shows the loyalty points

```
choose operation: 3
input customer name: chesson
chessontotal loyalty points: 1500
```

Input “4” Redeeming points

```
choose operation: 4
input customer name: chesson
input the points want to used for redemption: 2000
chessonhas done redemption by using loyalty points: 2000
```

Input “5” shows the loyalty status

```
choose operation: 5
input customer name: chesson
chessonloyalty status: high level
```

Input “6” reset the loyalty points to zero (0)

```
choose operation: 6
input customer name: chesson
```

Input “7” shows the points which have been reset

```
choose operation: 7
chesson: total loyalty points: 0, has done redemption: 0
```

Input “8” Exit the system

2.2 special cases and error messages

Some special cases for error solutions, due to the time limited, I won't demonstrate one by one, you can check detailedly in the source codes:

If the customer's name has existed

```
choose operation: 1
input customer name: chesson
input customer contact details: 66898
Customers already exist!
```

If the customer has no enough points

```
choose operation: 4
input customer name: chesson
input the points want to used for redemption: 1000
No enought loyalty points!
```

When user choose options, invalid input

```
choose operation: c
invalid input, please input again!
```