**Birthday Party Planning Management System**

**Group ID : MTRO\_G1\_06**

**Batch No : METRO G1**

**Campus : Metro Campus**

**Group Details:**

|  |  |  |
| --- | --- | --- |
|  | **Student Registration Number** | **Student Name** |
| **1** | **IT17050272** | **D. Manoj Kumar** |
| **2** | **IT14132506** | **K. Vithujan** |
| **3** | **IT17050722** | **R. Rahul Raj** |
| **4** | **IT17032766** | **K. Bavanraj** |
| **5** | **IT17030618** | **B. Sharounthan** |

**Date : 22/09/2017**

Birthday Party Planning Management System

1. Customer should register first and he must give his details such as name, e-mail id etc.
2. A customer must view free dates, halls and decoration types in options list.
3. As a customer, I have to make an order according to my convenience. (user story)
4. The total amount of the order can be viewed by the customer.
5. Customer can make payment with cash or credit card.
6. Customers have to view their booked items in orders section.
7. If the customer confirms the order it will be updated in the system.
8. Inventory officer must update item availability in decoration materials, halls and dates.
9. If there any material shortages inventory officer should informer manager.
10. Receptionist must view payment details.
11. According to the payment details receptionist can accept or reject order.
12. Receptionist should inform the customer if there are any issues.
13. As a cashier, I have to check payment details so that I can make reports. (User story)
14. Manager has to check the order list and the list of inventory items.
15. As the manager, I have to check reports so that I can make the annual report. (User story)

**Classes**

1. Customer
2. Options
3. Payments
4. Orders
5. Manager
6. Receptionist
7. Cashier
8. Inventory\_Officer

**CRC Cards**

|  |  |
| --- | --- |
| **Class name: Customer** | |
| **Responsibilities** | **Collaborations** |
| Register details | Orders |
| View options | Options |
| Make order | Payments |
| Make payment |  |

|  |  |
| --- | --- |
| **Class name: Options** | |
| **Responsibilities** | **Collaborations** |
| Show items availability |  |
| Update hall, free dates and decoration types |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Class name: Payments** | |
| **Responsibilities** | **Collaborations** |
| Register payment details | Customer |
| Display the cost for the orders | Orders |
| Validate payment |  |
|  |  |

|  |  |
| --- | --- |
| **Class name: Orders** | |
| **Responsibilities** | **Collaborations** |
| Update order details | Customer |
| Confirm order details | Options |
| Show booked items |  |
|  |  |

|  |  |
| --- | --- |
| **Class name: Manager** | |
| **Responsibilities** | **Collaborations** |
| Check order list and inventory items | Options |
| Generate annual report | Orders |
|  | Cashier |
|  |  |

|  |  |
| --- | --- |
| **Class name: Receptionist** | |
| **Responsibilities** | **Collaborations** |
| View payment details | Payments |
| Accept or reject order | Orders |
| Inform customer if any issues | Customer |
|  |  |

|  |  |
| --- | --- |
| **Class name: Cashier** | |
| **Responsibilities** | **Collaborations** |
| Check payment details | Payments |
| Generate report |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Class name: Inventory\_Officer** | |
| **Responsibilities** | **Collaborations** |
| Update item availability | Options |
| If any material shortage inform to manager | Manager |
|  |  |
|  |  |

**Classes**

Classes are programming templates which consist of data members and member functions to do some tasks.

Example: customer{

Properties: name

address

phone\_no

Methods: display\_details()

}

**Objects**

Objects are created to store data’s and to do some tasks. We can create many objects for a certain class. In our case we can create customer1, customer2 as the objects of the customer class. So that customer1 and customer2 can store different details.

**Properties**

Data which are inside a class is called as properties. We can simply say as variables or attributes. Name, phone\_no and address are some of the properties in the customer class. Here we can store the name, phone number and the address of the customer.

**Methods**

Methods are like functions which we use in structured programming. Methods can do certain jobs using the properties of their classes. Here we can make a method “total\_bill()” in payment class to calculate the total bill by using the data members of the class.

**Encapsulation**

Encapsulation is storing the properties and methods together. So that we can prevent outsiders from considering it. The user will get an interface. So that he can’t see the encapsulated part.

Example: customer

Private: name;

address;

phone\_no;

public: display\_details();

Here we have stored both properties and methods together in a class by preventing outsiders. This shows how the encapsulation is defined.

**Abstraction**

Abstraction in OOC means selecting the essential things. In payment class, we need some details like total payment, paying method and customer name. They are essential things to the class. So, we should add them. Payment class don’t need the customer phone number or address. So, we don’t have to include them. This process is called abstraction.

**Polymorphism**

Polymorphism is using the same name for different purposes. we can create more than one property, method or an object with the same name in different classes. For example, we can create “total()” function in both “payments” and “cashier” class.

**Inheritance**

Inheritance is used for code reuse. In inheritance concept, we can share some of the properties and methods from another class to another class. So, we don’t have to type the codes again. In our case “manager” class can inherit “cashier” class.