Case Study-1

Parking Lot

System Requirements (Excepted and can be added more)

1. The parking lot should have multiple floors where customers can park their vehicles.

2. The parking lot should have multiple entry and exit points.

3. Customers can collect a parking ticket from the entry points and can pay the parking fee at the exit points on their way out.

4. Customers can pay the tickets at the automated exit panel or to the parking attendant.

5. Customers can pay via both cash and credit cards.

6. Customers should also be able to pay the parking fee at the customer's info portal on each floor. If the customer has paid at the info portal, they don't have to pay at the exit.

7. The system should not allow more vehicles than the maximum capacity of the parking lot. If the parking is full, the system should be able to show a message at the entrance panel and on the parking display board on the ground floor.

8. Each parking floor will have many parking spots. The system should support multiple types of parking spots such as Compact, Large, Handicapped, Motorcycle, etc.

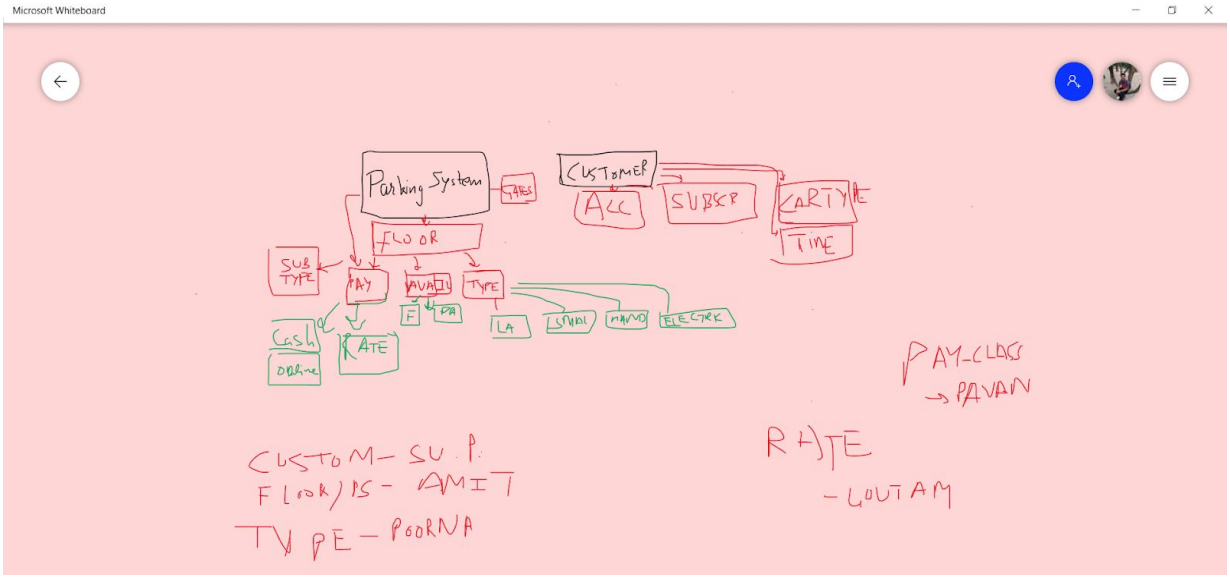
9. The Parking lot should have some parking spots specified for electric cars. These spots should have an electric panel through which customers can pay and charge their vehicles.

10. The system should support parking for different types of vehicles like car, truck, van, motorcycle, etc.

11. Each parking floor should have a display board showing any free parking spot for each spot type.
12. The system should support a per-hour parking fee model. For example, customers have to pay Rs 20 for the first hour, Rs 10 for the second and third hours, and Rs 5 for all the remaining hours.

CS19B003- AMIT KESARI : Worked on Solution.Java, ParkingLot.Java, CallMenu (Interface)
CS19B006-BAINDLA GOUTHAM : Worked on Bill/Rate Calculation
CS19B017-PAVAN SAHITH : Worked on Payment.Java
CS19B039-SUMAN P : Worked on Customer.Java
CS19B049-POORNA SYAMA SUNDAR : Worked on Floor.java , VehicleSlot.java

Rough Sketch:



Solution.java

- We have made Solution.java in which there is Admin Dash and Customer Dash.
- Admin Dash is for configuring the parking lot and setting up the floors.
- Customer Dash is for getting information from the customer.

Parkinglot.java

- In this class we are setting up the number of floors using ArrayList.
- We are also displaying customer details and individual floor details.

Floors.java

- Each individual floor is configured through here.

- We are setting up VehicleSlot of various types of vehicles using ArrayList.
- We have also included functions for booking slots and exiting the slot.

Payment.java

- It displays the bill and asks the customer to pay through various methods of payment.

Customer.java

- We get all the details from the customer such as username, vehicle type,account balance,staytime.
- Customer can book the slot on the floor he prefers.

He/She can also exit the parking lot after payment.

VehicleSlot.java

- Setting Slots of different vehicle types on specific floors.

CallMenu.java (Interface)

- An interface used for displaying the Menu driving for the program and invoking functions according to the input given by the user.

Discussion-1(dt:08-10-2020)

- Discussed about the basic layout of the Project.(A rough sketch of the plan which was discussed is provided above.)
- Assigned works to each member.
- Decisions made in the Discussion:
    - Two types of users are defined Admin and Customer.
    - The number of floors and parking slots are dynamic and can be changed by Admin
    - Decided to divide the work of payment, floors, slots, and customer into different classes to emphasize **Object Oriented programming**.(The complete functionality of each class is provided above.)
    - One interface (CallMenu) should be implemented by the classes Floors, ParkingLot, Payment and gets extended to Customer through ParkingLot to emphasize **Inheritance**, **Abstraction** and **Polymorphism (**as the methods of CallMenu are overridden in each class they are implemented).
    - As the classes VehicleSlot, Floors are used as objects they aren't declared as interfaces.
    - All the variables should be private and can be accessed only through public get and set methods to provide an additional layer of security and also to see that there is no problem of assigning wrong or irrelevant values.

- Decided to have a discussion after midsem.

Discussion-2(dt:17-10-2020)

- Decided to have different base rates for different Vehicle Types.
- Made minor corrections and improvements in the plan.
- Completed almost 90% of the code.

Discussion-3(dt:22-10-2020)

- Added password for Admin.
- Ran the entire code for the first time.
- Key takeaways after running and testing the code.
    - Decided to have a minimum no of floors and slots and an option to further change that (instead of asking the admin to initialize the no of floors and slots each time the code is run)
    - Realized that the bill is not attributed to the customer and made changes so that the bill is fixed for each customer and doesn't have new value each time the payment class is called.
    - Added vehicle slot for electric vehicles { this wasn't noticed in the first discussion}
    - Assigned Id's for each vehicle Slot.