Github repo link

Note:
Please use this doc to note down your discussions and ideas related to the case study. This doc will be used during the evaluation along with the source code.
After deciding all the things related to the case study, mention the contribution from each team member.

Team Members Name:
CS20B023 Karthik
CS20B030 Preethi Varsha Marivina
CS20B037 Sarthak Girotra
CS20B029 Pranav Sutar

Few things that you can note here:
How to design the system?
How many classes do we need?
Any new things we might have to learn while developing the system?
Who is doing what works?
What things are we adding that are not mentioned in the case study instructions?

1. Entry and Exit time is automatically calculated .
2. Facility to configure the number of floors, slots and prices for each slot.
3. Fastag and UPI payment options.
4. Incase customer is short of money, Credit card payment is done.

What type of variables, functions might we need to use?
What is the final design of the system (agreed by all)? (you can boxes and show the connections between them)

Objectives for PM_Parking_LOT_Team_4

1. Provide a menu at the entry point where customers can provide info about their vehicle
    a. 2 wheeler or 4 wheeler
    b. Type of vehicle (based on 2 or 4 wheeler)
    c. For both 2 or 4 wheeler - electric or fuel
    d. Displaying which slots are available - Compact, Large, Handicapped, Motorcycle, etc. (based on vehicle type)
    e. Customers can choose the free slot
2. Charge to be paid at the parking slot or exit - cash, Fastag,memberships, credit and UPI(can be extended more) option.
3. A function for Displaying AND DIRECTING towards vacant Slots at ticket collecting centers for a particular type of vehicle.
4. A function for content displaying on every floor

5. Method for payment calculating for parking, and also for charging electric vehicles
6. Assign a ticket to each customer which should provide info about the floor level, vehicle type, and the index of the parking slot being occupied(denoted by a string)
7. Electric cars-
    a. Implement a panel for electric vehicles which shows details of the car
    b. Make a charging cost panel for the car
8. Electric motorcycles(same facilities as electric cars)

Classes being used

1. **Payment**
2. **Manager**
3. **Menu**
4. **Customer**
5. **parkingSlot**
6. **Slots**
7. **Timer_Class**

Manager(so that parking lot customization can be done)

- Decide the number of floors and number of parking slots and types of slots for each floor.
- Decide the cost for each parking slot.
- ★ Rationale:
    - ➔ To keep the system Flexible- Large or Small
    - ➔ To Modify the Slot capacity according to requirement.
    - ➔ To change the cost of charging according to inflation and availability.

Menu

- Displays the options available to the customer and also connects with the customer class to store all the information.
- Take input - “New Customer” or “Existing Customer” or “Exit Program”
    a. **Existing Customer**
        i. Note exit time(automated)
        ii. Display exit options (which gate) and whether payment is done to the attendant or not.
        iii. Display cost of parking based on customer details and remove the customer from the list
    b. **New Customer:**
        i. Create Customer data
        ii. Type of vehicle
        iii. Input Electric or diesel
        iv. Display free slots
        v. Input floor and slot choice from the customer
        vi. Input Entry gate (Gate 1 or 2)
        vii. Note entry time (automated)

viii. Ask payment method
ix. Display empty floor and slots and assign slot in the chosen floor.

c. Exit Program:

i. Terminates program

★ Rationale:

➔ This class is required to bring out the user interface.

➔ To collaborate different activities by same customer without messing up with the data of other customers

Main Block

- Boolean Array [floor] [slots in each floor] - linked to manager functions -> floor value and number of slots.
-

Parking slot → interface

- Being implemented by parkingSlot

Payment

- Implements the *Timer_Class* interface
- Based on the vehicle type
  - Electric- two types of prices charging fees and the parking fees
  - Normal- only the parking fees

- From parking, class retrieve the info
  Rationale:
  ➔ A system is required to calculate the fees

Timer_Class → interface

- Used as an interface which will be implemented by the payment class
- It contains the method to calculate the total time spent by a vehicle in the parking slot and that time is being taken from the system clock.
- This total time is then being used to calculate the total charge in the Payment class

★ Rationale:

➔ To automatically know for what duration was the vehicle parked

➔ To sync the code with the clock of the computer.

Slots

- Contains slot allocation and removal methods.
- Assign slots to according to the type of vehicle registered by the customer
- The slots are assigned on the basis of the earliest free slot available in the list

Customer

- Contains all the instance variables to store the information of the user,vehicle,payment etc.
- This class is also being used to extract all the information entered earlier by the user
- Ex: userName, payment method, floor number, slot, type of vehicle, electric or diesel, still parked or departed.
- An object of this class can be used as a ticket.
- This class also includes error handling(if the user enters a wrong input then an exception will be thrown)
- ★ Rationale:
  - ➔ To identify each user separately
  - ➔ We can make Java list of this class
  - ➔ While exiting, we and track the history of the customer (eg. entry time, vehicle type,etc.) to calculate the payment

Ideas not used (Discarded)

- Pre Deciding the number of floors and the number of parking slots → this access was given to the manager
- User had to input the entry and exit time but then it was automated in the *Timer_Class* interface
- We Decided Floor Class, but then rejected it and made a *Slots* class .
- An interface which would be implemented by the classes for different types of vehicles was thought of first but was eventually dropped because the same thing could be done by using *parkingSlot* and *Slots* class which could take into account the type of vehicles

Assumptions made

- The parking lot configuration can only be set by the manager and only at the start of the program and not in between(a default configuration has also been set up for the case where manager does not set up the lot)
- The different types of slots that have been taken into account are
  - Slots for handicapped

        - Electric
        - Diesel/Petrol/CNG
    - For motorcycles(2 wheelers)
        - Electric
        - Diesel/Petrol/CNG
    - Compact(cars,vans,auto-rickshaws)
        - Electric
        - Diesel/Petrol/CNG
    - Large(buses,trucks etc)
        - Electric
        - Diesel/Petrol/CNG\

- Slots being allotted to the customer is the earliest available slot in the list of slots for the particular vehicle type.
- 1 second in real life= 1hour for the program(fees calculations have been made using this assumptions)
- The charging time for electric vehicles has been taken from the user

Contribution by each Member:

1)Sarthak - Slots,Menu,document
2)Preethi - Timer_Class, Payment, Customer,parkingSlot,document
3)Pranav - Manager_Class, Floor Layout(Miro), PPT ,document
4)Kartik - document