รายงาน

เรื่องระบบลานจอดรถ (Parking Lot)

อาจารย์ผู้สอน

ผศ.ธนา หงษ์สุวรรณ

จัดทำโดย

นายธนนนท์ สมบูรณ์ รหัสนักศึกษา 64015057

นายนฤนาถ ตันตื้อ รหัสนักศึกษา 64015068

นายพิสิฐพงศ์ พิสิฐแก้วเพชร รหัสนักศึกษา 64015102

นายรนวิทย์ เหมือนแก้ว รหัสนักศึกษา 64015117

ภาควิชาวิศวกรรมคอมพิวเตอร์(ต่อเนื่อง)

รายงานเล่มนี้เป็นส่วนหนึ่งของวิชาราย

01076105 Object Oriented Programming

ภาคเรียนที่ 2 ปีการศึกษา 2564

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Design a Parking Lot

ที่จอดรถเป็นพื้นที่โล่งสำหรับจอดรถโดยเฉพาะ ในประเทศส่วนใหญ่ที่รถยนต์เป็นพาหนะหลัก ลานจอดรถเป็นจุดเด่นของทุกเมืองและเขตชานเมือง ห้างสรรพสินค้า สนามกีฬา โบสถ์ใหญ่ และสถานที่ที่คล้ายกันมักมีที่จอดรถในพื้นที่ขนาดใหญ่

System Requirements

ความต้องการของระบบ

จะมุ่งเน้นไปที่ข้อกำหนดต่อไปนี้ในขณะที่ออกแบบที่จอดรถ:

1. ที่จอดรถควรมีหลายชั้นซึ่งลูกค้าสามารถจอดรถได้
2. ที่จอดรถควรมีทางเข้า/ออกหลายจุด
3. ลูกค้าสามารถรับบัตรจอดรถได้จากจุดทางเข้า และสามารถชำระค่าจอดรถที่จุดทางออกได้
4. ลูกค้าสามารถชำระตั๋วได้ที่แผงทางออกอัตโนมัติหรือที่พนักงานจอดรถ
5. ลูกค้าสามารถชำระเงินได้ทั้งเงินสดและบัตรเครดิต
6. ลูกค้าควรสามารถชำระค่าจอดรถได้ที่แผงอัตโนมัติของลูกค้าในแต่ละชั้น หากลูกค้าชำระเงินที่แผงอัตโนมัติแล้ว ก็ไม่ต้องจ่ายกับพนักงานที่ทางออก
7. ระบบไม่ควรให้รถเกินความจุสูงสุดของที่จอดรถมาจอด หากที่จอดรถเต็ม ระบบควรจะสามารถแสดงข้อความที่แผงทางเข้าและ บนแผงแสดงผลที่จอดรถที่ชั้นล่างได้
8. ที่จอดรถแต่ละชั้นจะมีที่จอดรถหลายจุด ระบบควรรองรับที่จอดรถได้หลายประเภท เช่น ขนาดเล็ก ขนาดใหญ่ คนพิการ รถจักรยานยนต์ เป็นต้น

9. ที่จอดรถควรมีจุดจอดรถสำหรับรถยนต์ไฟฟ้าโดยเฉพาะ
จุดเหล่านี้ควรมีแผงไฟฟ้าที่ลูกค้าสามารถชำระเงินและชาร์จรถยนต์ได้

10. ระบบควรรองรับการจอดรถสำหรับยานพาหนะประเภทต่างๆ เช่น รถยนต์ รถบรรทุก รถตู้ รถจักรยานยนต์ เป็นต้น

11. ที่จอดรถแต่ละชั้นควรมีแผงป้ายแสดงว่ามีที่ว่างกี่ที่ ของรถแต่ละประเภท

12. ระบบควรรองรับรูปแบบค่าจอดรถรายชั่วโมง ตัวอย่างเช่น ลูกค้าต้องจ่าย $4 สำหรับชั่วโมงแรก, $3.5 สำหรับชั่วโมงที่สองและสาม และ $2.5 สำหรับชั่วโมงที่เหลือทั้งหมด

Use Case Diagram

นี่คือนักแสดงหลักในระบบ :

- Admin : รับผิดชอบหลักในการเพิ่มและแก้ไขชั้นที่จอดรถ, จุดจอดรถ, แผงทางเข้าและทางออก, การเพิ่ม/ถอด ผู้ดูแลที่จอดรถ(Parking Attendant) ฯลฯ

- Customer : ลูกค้าทุกคนสามารถรับบัตรจอดรถและชำระเงินได้

- Parking Attendant : พนักงานจอดรถสามารถทำกิจกรรมทั้งหมดในนามของลูกค้า และสามารถรับเงินสดเพื่อชำระค่าตั๋วได้

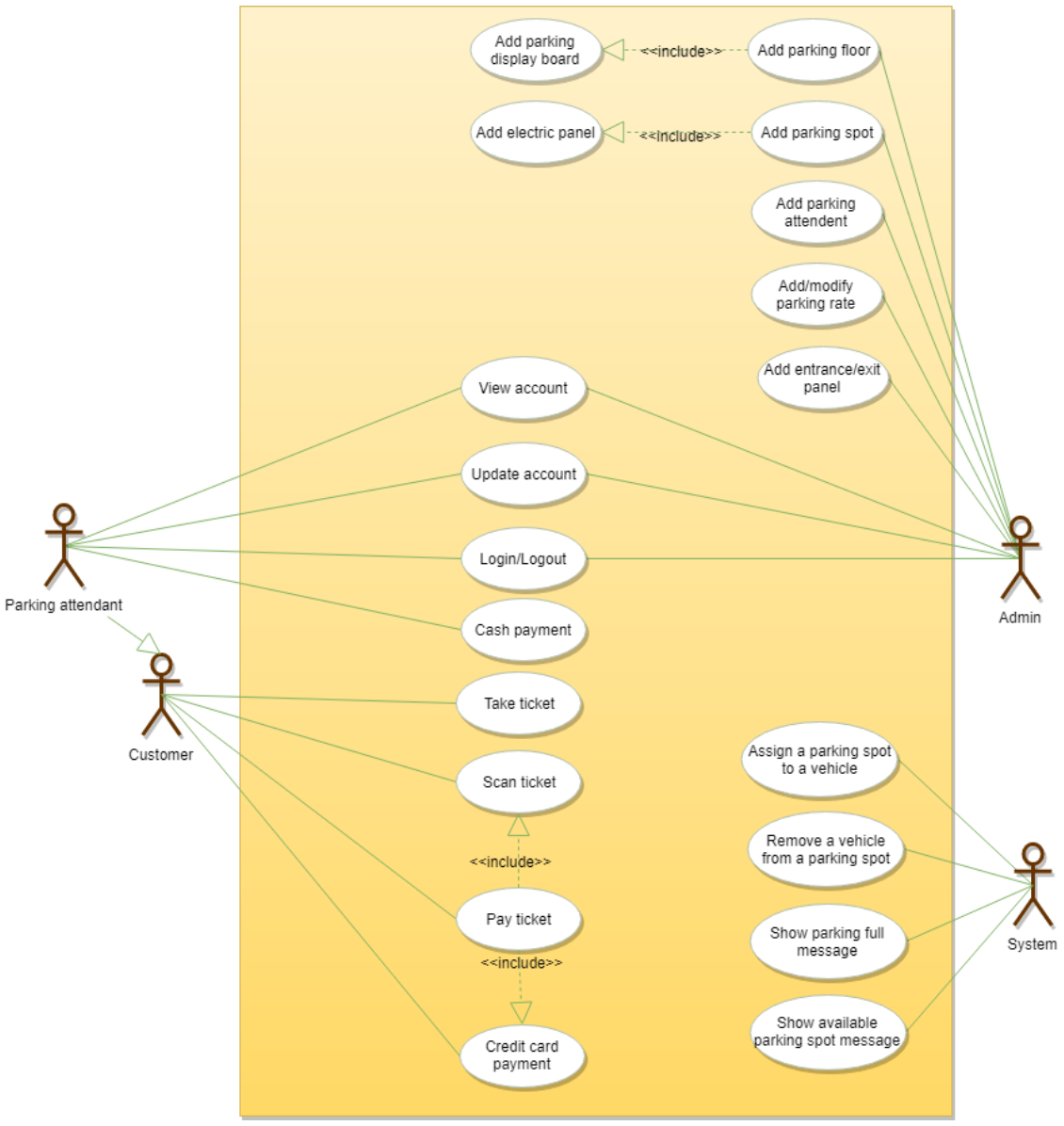
- System : เพื่อแสดงข้อความบนแผงข้อมูลต่างๆ รวมทั้งกำหนดและนำรถออกจากจุดจอดรถ

ต่อไปนี้คือกรณีการใช้งานยอดนิยมสำหรับที่จอดรถ :

- Add/Remove/Edit parking floor : การเพิ่ม ลด หรือปรับเปลี่ยนพื้นที่จอดรถออกจากระบบ แต่ละชั้นสามารถมีบอร์ดแสดงผลของตัวเองเพื่อแสดงจำนวนจุดจอดรถที่ว่างอยู่

- Add/Remove/Edit parking spot : หากต้องการเพิ่ม ลบ หรือแก้ไขจุดจอดรถบนพื้นที่จอดรถ

- Add/Remove a parking attendant : การเพิ่มหรือลบผู้ดูแลที่จอดรถ(Parking Attendant) ออกจากระบบ
- Take ticket : เพื่อให้ลูกค้าได้รับบัตรจอดรถใหม่เมื่อเข้าสู่ที่จอดรถ
- Scan ticket : เพื่อคำนวณหาค่าใช้จ่ายทั้งหมด
- Credit card payment : เพื่อชำระค่าธรรมเนียมบัตรโดยสารด้วยบัตรเครดิต
- Cash payment : เพื่อชำระค่าจอดรถด้วยเงินสด
- Add/Modify parking rate : เพื่อให้แอดมิน(Admin)เพิ่มหรือแก้ไขอัตราค่าจอดรถรายชั่วโมง



Class diagram

ระบบหลักของที่จอดรถ

1. ParkingLot: ส่วนกลางของระบบที่ซอฟต์แวร์นี้ ออกแบบให้ มีคุณลักษณะ เช่น 'ชื่อ' เพื่อแยกความแตกต่างจากที่จอดรถอื่น ๆ และ 'ที่อยู่' เพื่อกำหนดตำแหน่ง

2. ParkingFloor: ลานจอดรถมีหลายชั้น

3. ParkingSpot: ระบบรองรับที่จอดรถหลายประเภท ได้แก่

    1.) ผู้พิการ (Handicapped) 2.) รถเล็ก(Compact) 3.) รถใหญ่(Large)

    4.) รถมอเตอร์ไซค์(Motorcycle) 5.) รถไฟฟ้า(Electric)

4. Account: มี 2 account

    1.) สำหรับผู้ดูแล (Admin)

    2.) สำหรับพนักงานจอดรถ (Parking attendant)

5. Parking ticket: ลูกค้าจะรับบัตรเมื่อเข้าที่จอดรถ

6. Vehicle: ระบบสนับสนุน

    1.) รถยนต์ (Car) 2.) รถบรรทุก (Truck) 3.) รถไฟฟ้า (Electric) 4.) รถตู้ (Van)

    5.) รถมอเตอร์ไซค์ (Motorcycle)

7. EntrancePanel and ExitPanel:

    - EntrancePanel แผงทางเข้า จะพิมพ์ตั๋ว

    - ExitPanel แผงทางออก จะอำนวยความสะดวกในการชำระค่าธรรมเนียมตั๋ว

8. Payment: คลาสนี้ รับผิดชอบในการชำระเงิน ระบบจะสนับสนุน บัตรเครดิต และการโอนเงินสด

9. ParkingRate: คลาสนี้ จะติดตามอัตราการจอดรถทุกชั่วโมงจะระบุจำนวนเงินดอลลาร์สำหรับแต่ละ

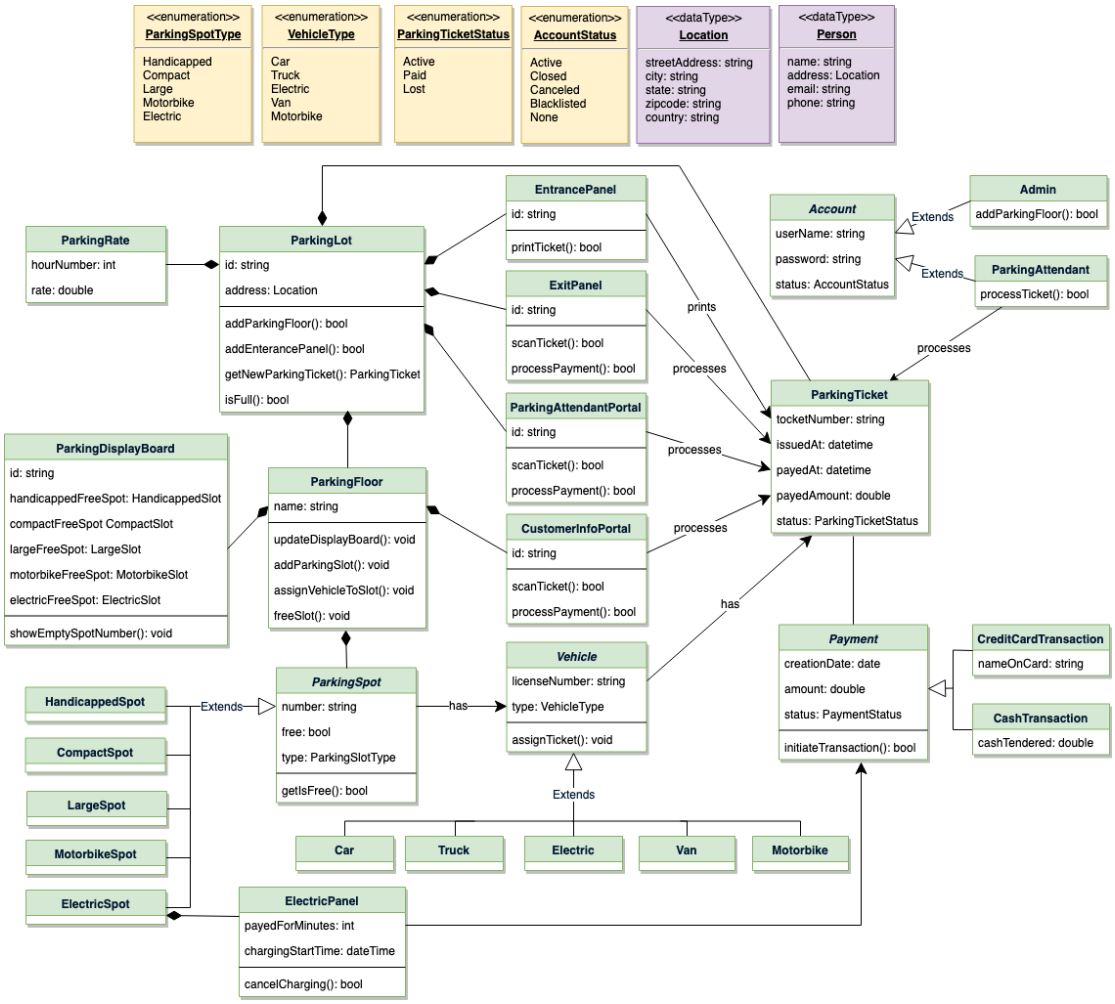
ชั่วโมงตัวอย่างเช่น บัตรจอดรถสองชั่วโมง คลาสนี้จะกำหนดค่าใช้จ่ายสำหรับชั่วโมงแรก และชั่วโมงที่สอง

10. ParkingDisplayBoard: แต่ละชั้นที่จอดรถจะมีบอร์ดแสดงผล
เพื่อแสดงจุดจอดรถแต่ละจุดคลาสนี้จะเป็นผู้รับผิดชอบในการแสดงจุดจอดที่ว่างอยู่ให้กับลูกค้า

11. ParkingAttendantPortal: คลาสนี้ จะครอบคลุมการดำเนินงานทั้งหมด ที่พนักงานสามารถดำเนินการได้ เช่น การสแกนตั๋ว และการชำระเงิน

12. CustomerInfoPortal: ในคลาสนี้ จะเข้าถึงข้อมูล ที่ลูกค้าใช้จ่ายสำหรับบัตรจอดรถ เมื่อจ่ายแล้วประตูข้อมูลจะอัปเดตตั๋วเพื่อติดตามการจ่ายเงิน

13. ElectricPanel: ลูกค้าจะใช้แผงไฟฟ้า เพื่อชำระเงินและชาร์จรถยนต์ไฟฟ้า



หน้าที่การรับผิดชอบ

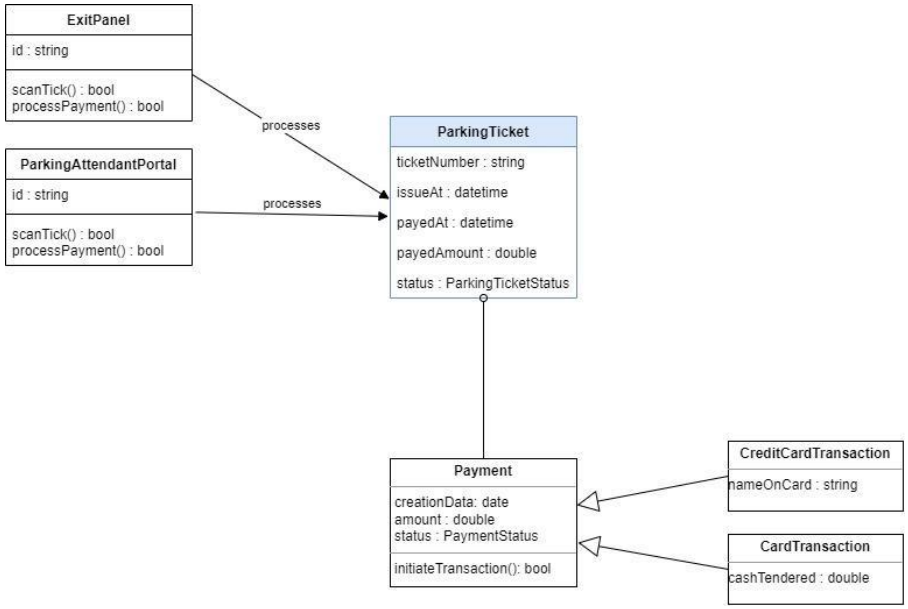
รหัสนักศึกษา	ชื่อ-นามสกุล	Class ที่รับผิดชอบ
64015057	นายธนนนท์ สมบูรณ์	Take ticket
64015068	นายนฤนาถ ตันตื้อ	Cash payment
64015102	นายพิสิฐพงศ์ พิสิฐแก้วเพชร	Add/Remove/Edit parking floor
64015117	นายรนวิทย์ เหมือนแก้ว	Add/Remove/Edit parking spot

Use Case Description ของ Cash payment

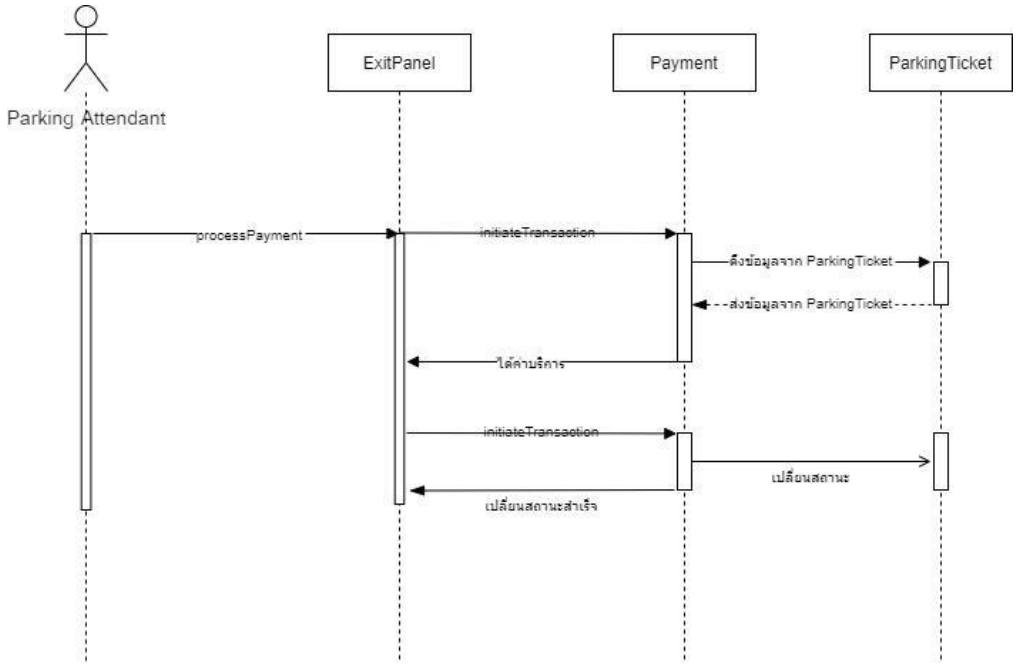
Name	Cash payment
Participating actor	Parking Attendant
Entry condition	-เมื่อลูกค้า(Customer) อยู่ที่จุดชำระเงิน (ExitPanel) -เมื่อลูกค้า(Customer) อยู่ที่จุดชำระเงินพนักงาน (ParkingAttendantPanel) -ต้องมีตั๋ว
Exit condition	จ่ายเงินสำเร็จและค่าstatusเปลี่ยนแปลง
Event flow	-เมื่อลูกค้า(Customer) อยู่ที่จุดชำระเงิน (ExitPanel) -จะใช้ฟังก์ชั่น initiateTransaction -จากนั้นดึงข้อมูลจาก ParkingTicket -จากนั้นดึงข้อมูลจาก ParkingTicket เพื่อตรวจสอบค่าบริการ -แจ้งค่าบริการกับลูกค้า -ลูกค้าจ่ายเงิน -เปลี่ยนค่าสถานะใน ParkingTicket

Exceptional Case	-ไม่มีเงิน เงินไม่พอ -สกุลเงินที่มีไม่ใช่ $

Class Diagram ของ Cash payment



Sequence Diagram ของ Cash payment



Code ของ Cash payment

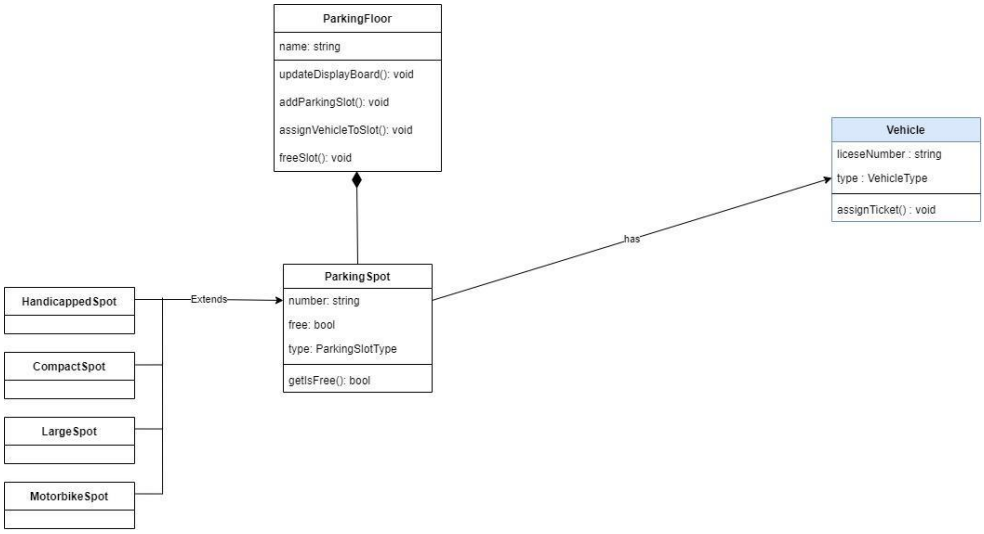
```
class ExitPanel:

    def __init__(self, id) :
        self.id = id

    def scanTick():
        pass

    def processPayment():
        pass

class ParkingAttendantPortal:

    def __init__(self, id) :
        self.id = id

    def scanTick():
        pass

    def processPayment():
        pass


class Payment:

    def __init__(self, creationData, amount, status) :
        self.creationData = creationData
        self.amount = amount
        self.status = status

    def initiateTransaction():
        pass

class CreditCardTransaction(Payment):

    def __init__(self, creationData, amount, status, nameOnCard):
        super().__init__(creationData, amount, status)
        self.nameOnCard = nameOnCard

class CardTransaction(Payment):

    def __init__(self, creationData, amount, status, cashTendered):
        super().__init__(creationData, amount, status)
        self.cashTendered = cashTendered
```

Use Case Description ของ Add/Remove/Edit parking spot

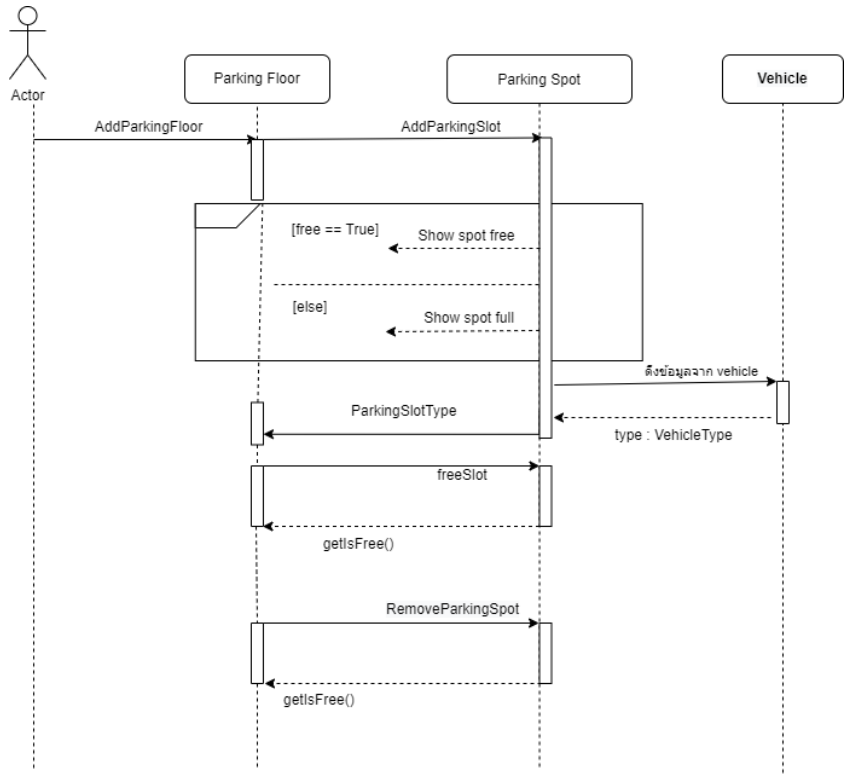
Name	Add/Remove/Edit parking spot
Participating actor	Admin
Entry condition	- ต้องมีพื้นที่จอดรถ - มีเหตุการณ์เช่น ลูกค้านำรถมาจอด หรือ นำรถออกจากที่จอด
Exit condition	- ได้จุดจอดรถ บนพื้นที่จอดรถ
Event flow	- ยูสเคสเริ่มต้นเมื่อ มี Customer เข้าที่จอดรถ - ดึงข้อมูลมาจาก Parking Floor - ใส่ number ของประเภทของรถ ที่จะต้องการ add - remove จุดจอดรถออก เมื่อรถออกจากจุดจอดรถ

	- update parking spot (ส่งค่า จำนวนจุดจอดรถที่ว่าง) - End use case
Exceptional Case	- พื้นที่จอดรถเต็ม ทำให้ไม่สามารถ add จุดจอดรถเพิ่มได้

Class Diagram ของ Add/Remove/Edit parking spot



Sequence Diagram ของ Add/Remove/Edit parking spot



Code ของ Add/Remove/Edit parking spot

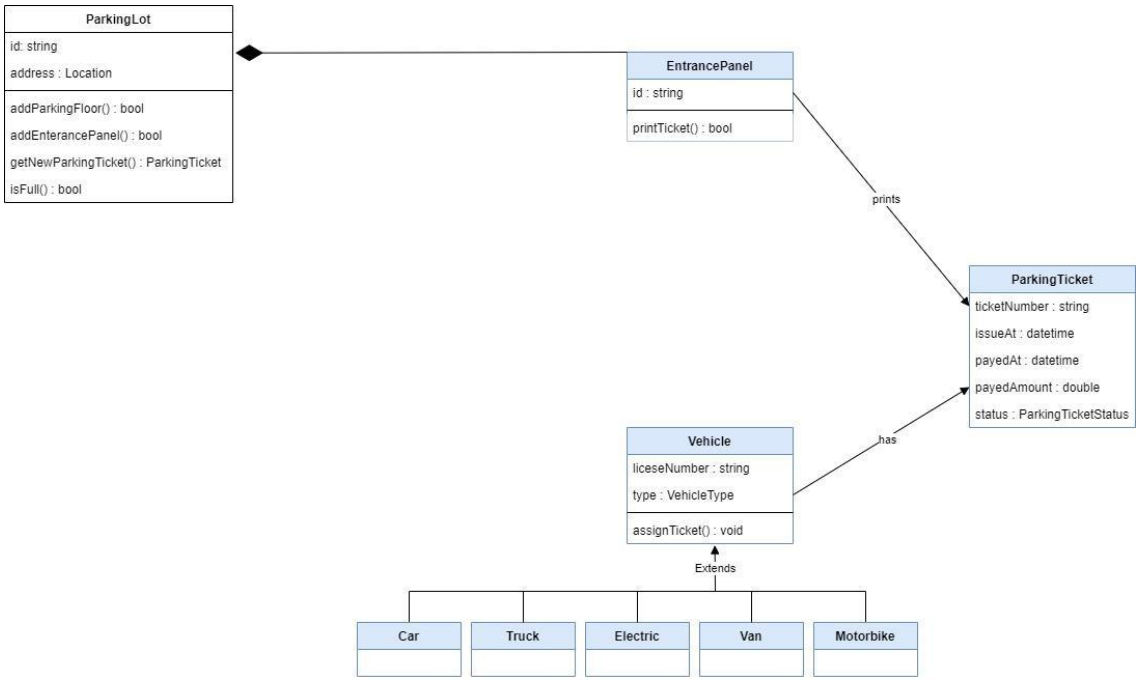
```python
from enum import Enum
from abc import ABC

class ParkingSpotType(Enum):
    HANDICAPPED, COMPACT, LARGE, MOTORBIKE = 1, 2, 3, 4

class ParkingSpot(ABC):
    def __init__(self, number, parking_spot_type):
        self.__number = number
        self.__free = True
        self.__vehicle = None
        self.__parking_spot_type = parking_spot_type

    def is_free(self):
        return self.__free

    def assign_vehicle(self, vehicle):
        self.__vehicle = vehicle
        free = False

    def remove_vehicle(self):
        self.__vehicle = None
        free = True

class HandicappedSpot(ParkingSpot):
    pass

class CompactSpot(ParkingSpot):
    pass

class LargeSpot(ParkingSpot):
    pass

class MotorbikeSpot(ParkingSpot):
    pass
```

Use Case Description ของ Take ticket

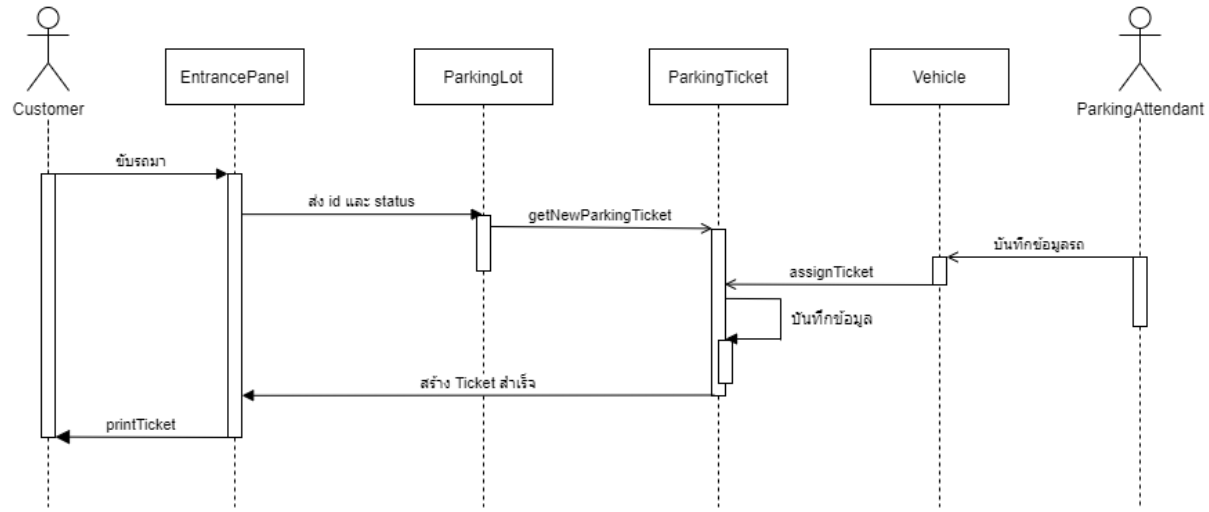
Name	Take ticket
Participating actor	Parking Attendant / Customer
Entry condition	-ลูกค้า(Customer)ต้องขับรถมาอยู่หน้าทางเข้า(Entrance Panel) -ทางเข้า(EntrancePanel)สร้างตั๋ว(ParkingTicket) -ตั๋ว(ParkingTicket)แบ่งประเภทตามพาหนะ(Vehicle)
Exit condition	-ลูกค้า(Customer)ได้รับตั๋ว(ParkingTicket)
Event flow	-เมื่อลูกค้า (Customer) ขับรถมาที่ทางเข้า (EntrancePanel) -ทางเข้าจะส่ง id และ status ไปยังที่จอดรถ (ParkingLot) -เรียกใช้ฟังก์ชัน getNewParkingTicket() -พนักงานจอดรถ(Parking Attendant)จะบันทึกข้อมูลทะเบียนรถและประเภทรถระบุลงไปในข้อมูลรถ

	(Vehicle) -เรียกใช้ฟังก์ชัน assignTicket() เพื่อบันบันทึกข้อมูลไปยังตั๋ว(ParkingTicket) เพื่อสร้าง Ticket -ทางเข้า (EntrancePanel) จะเรียกใช้ฟังก์ชัน printTicket() -ลูกค้า (Customer) ได้รับตั๋ว(Ticket)
Exceptional Case	-

Class Diagram ของ Take ticket



Sequence Diagram ของ Take ticket



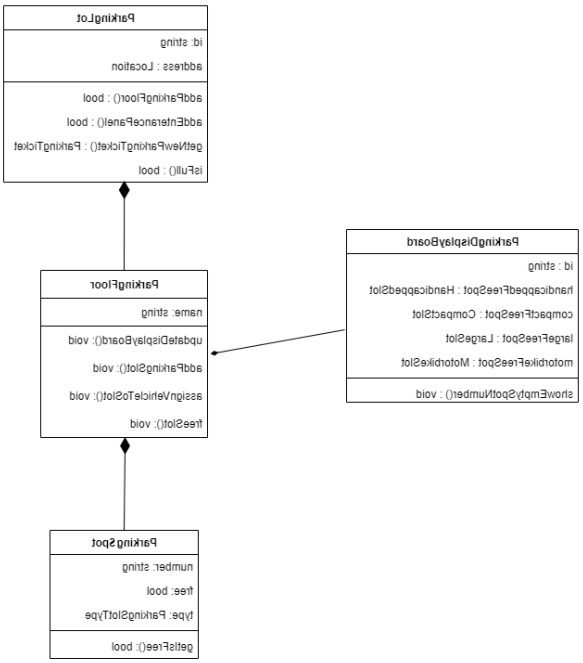
Code ของ Take ticket

```python
import enum

class EntrancePanel:
    def __init__(self, id):
        self.id = id

    def printTicket():
        pass

class ParkingTicketstatus (enum.Enum):
    Active = 0
    Paid = 1
    Lost = 2

class ParkingTicket:
    def __init__(self, ticketNumber, issuedAt, payedAt, payedAmount, status):
        self.ticketNumber = ticketNumber
        self.issuedAt = issuedAt
        self.payedAt = payedAt
        self.payedAmount = payedAmount
        self.status = status

class VehicleType(enum.Enum):
    Car = 1
    Truck = 2
    Electric = 3
    Van = 4
    Motorbike = 5

class Vehicle:
    def __init__(self, licenseNumber, type):
        self.licenseNumber = licenseNumber
        self.type = type

    def assignTicket():
        pass

class ParkingLot:
    def __init__(self, id, address):
        self.id = id
        self.address = address

    def addParkingFloor():
        pass

    def addEntrancePanel():
        pass

    def getNewParkingTicket():
        pass

    def isFull():
        pass
```

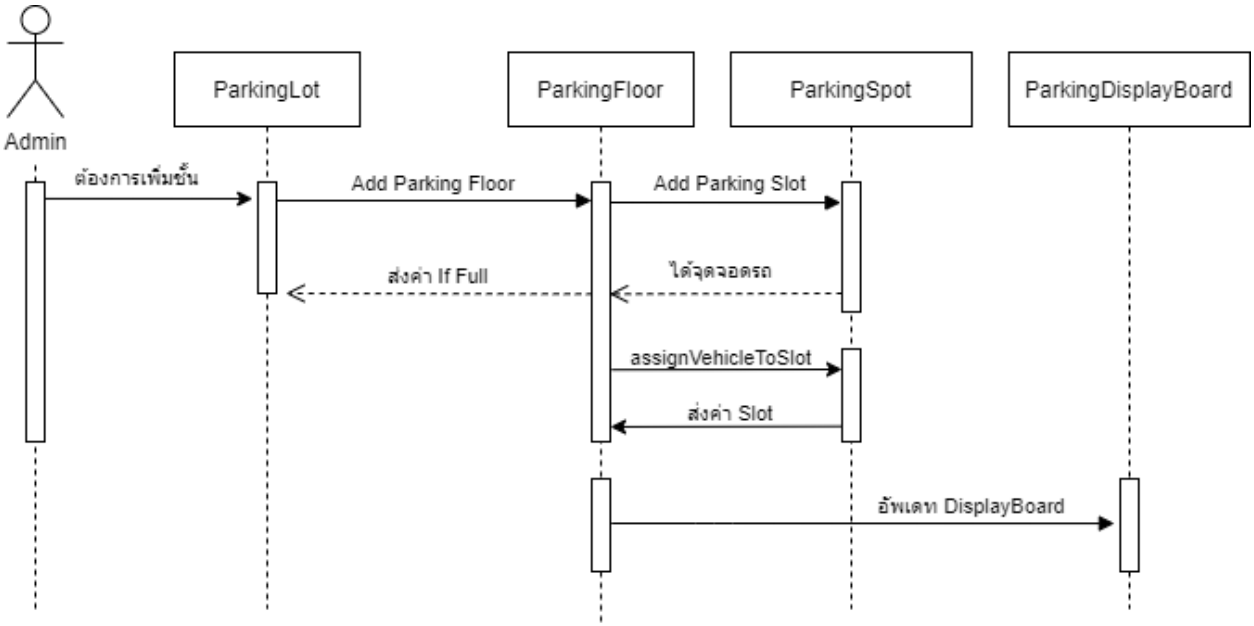
Use Case Description ของ Add/Remove/Edit parking floor

Name	Add/Remove/Edit parking floor
Participating actor	Admin
Entry condition	ต้องมีพื้นที่เพื่อสร้างชั้น
Exit condition	ได้ชั้นจอดรถเพิ่ม
Event flow	- เมื่อ Admin เรียกใช้ Parking floor - จะดึงข้อมูลจาก Parking Lot - ใส่ name ของชั้น ที่จะต้องการ Add/Remove/Edit - อัพเดทบอร์ดแสดงผล - แสดงจุดจอดรถที่ว่างที่บอร์ด
Exceptional Case	พื้นที่เต็ม ทำให้สร้างชั้นจอดรถเพิ่มไม่ได้

Class Diagram ของ Add/Remove/Edit parking floor



Sequence Diagram ของ Add/Remove/Edit parking floor



Code ของ Add/Remove/Edit parking floor

```python
class ParkingLot:

    def __init__(self, id, address):
        self.id = id
        self.address = address

    def addParkingFloor():
        pass

    def addEntrancePanel():
        pass

    def getNewParkingTicket():
        pass

    def isFull():
        pass

class ParkingFloor:

    def __init__(self, name):
        self.name = name

    def updateDisplayBoard():
        pass

    def addParkingSlot():
        pass

    def assignVehicleToSlot():
        pass

    def freeSlot():
        pass


class ParkingDisplayBoard:

    def __init__(self, id, handicappedSpot, compactSpot, largeSpot, motorbikeSpot) :
        self.id = id
        self.handicappedFreeSpot = handicappedSpot
        self.compactFreeSpot = compactSpot
        self.largeFreeSpot = largeSpot
        self.motorbikeFreeSpot = motorbikeSpot

    def showEmptySpotNumber() :
        pass
```