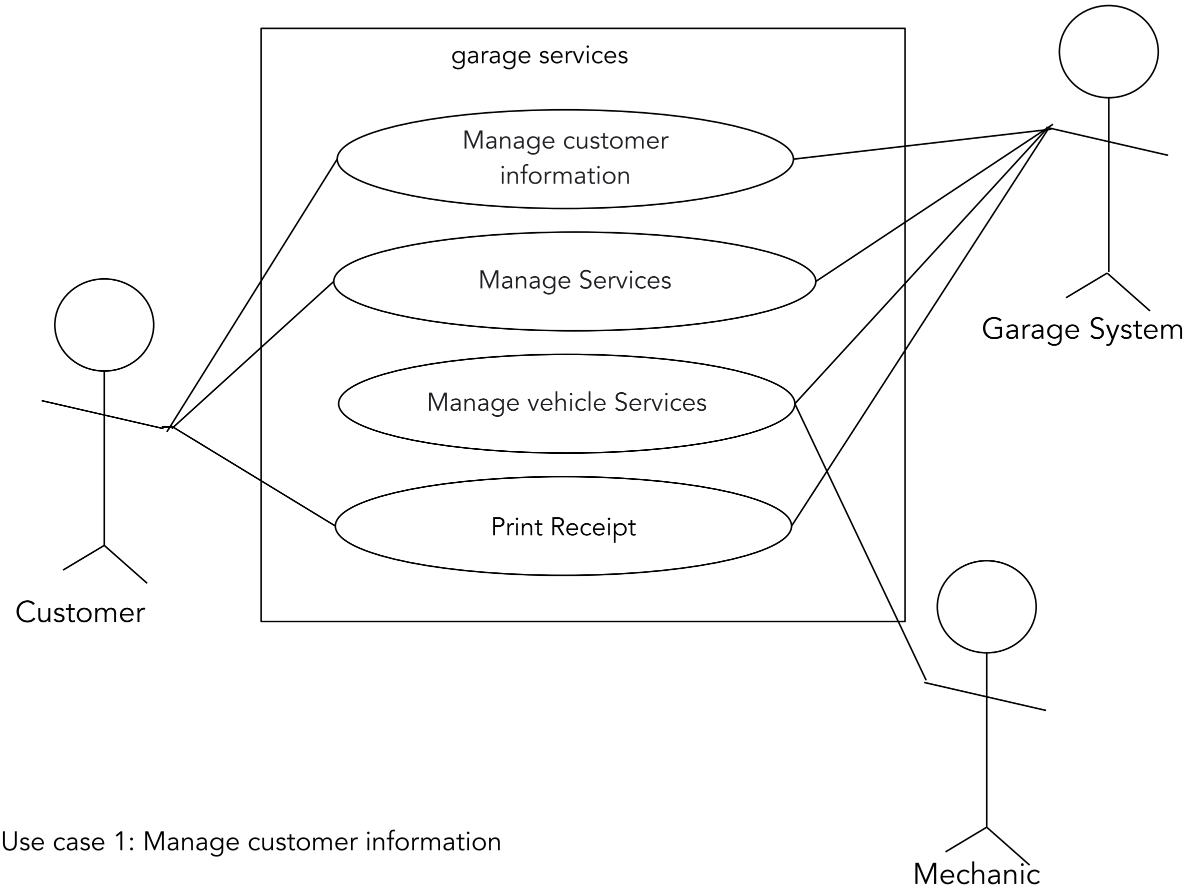
1. Identify the use cases for the software. Draw the **UML use-case diagram **and include supporting use-case descriptions. At-least 3 scenarios must be identified.

Use cases:



Use case 1: Manage customer information

Use Case:	Manage customer information
Trigger:	The system manages customer information and registers him/her in the system.
Main scenarios:	1. The customer specifies his/her personal information first name, last name, EID,email and phone number.
	2. The garage system verifies the customer's information
	3. The garage system asks the customer to provide the vehicle information, make,year,model, type, color, and ID.
	4. The garage system verifies the vehicle information.
2a	- The customer information is incorrect.

	- Ex. (The customer ID is invalid, The customer number is invalid…) - The use case ends, and an error message is communicated.
3a	- The vehicle information is incorrect. - The use case ends, and an error message is communicated

Use Case:	Manage Services
Trigger:	The customer wants to add a garage service.
Prediction:	-
Main scenario:	1. Customer Choose Manage Services from the main menu.
	2. The garage system displays a list of existing services.
	3. The customer selects the wanted services from the list.
	4. the customer can choose the unwanted serves and remove them from the list.
	5. The garage system updates the list.
	6. The garage system verifies the list
	8. The garage system gives the customer date to bring the car.
	9. The garage system verifies the date.
6a	- The user wants to add one extra service. - The user wants to remove the service. - The use case ends, and an error message is communicated.
9a	- If the date is not suitable for the customer. - If the mechanic is not available at that date. - The use case ends, and an error message is communicated.

Use Case:	Manage vehicle Services

Trigger:	- A customer brings a vehicle for service
Prediction:	- The customer is registered - A list of wanted services is generated.
Main scenario:	1. The garage system initiates a new service request for the customer's car.
	2. The garage system the assigns to the vehicle ID.
	3. A mechanic is sent to the service request by the garage system .
	4. The mechanic checks the service request.
	5. The mechanic performs the requested service on the vehicle.
	6. The mechanic logs the accomplished service with his name in the system.
	7. The mechanic confirm the service and tell the garage system that the service has been completed and write a brief description of the service and how long it takes with the price .
	8. The garage system reviews the completed service request.
	9. The garage system tells the customer that the vehicle is ready for pickup.
8a	- if there is a service that has not been accomplished the use case ends, and an error message is communicated.

Use Case:	Print Receipt
Trigger:	- The customer wants to pay for the services that have been done.
Prediction:	- A list of the done services is generated with the mechanic's name.
Main scenario:	1. The system tells the user to enter any discount code that is available.
	2. User enters discount code, if available.
	3. If the code is valid, the system takes the discount off the total.
	4. System adds taxes based on the total cost.

	5. The system shows the user the total cost with taxes and any discounts.
	6.the customer chose the payment method
	7.the garage system vafify the payment method .
	8.The system prints the receipt: - The customer information - The mechanic name - The vehicle information - The The date - The services with the price - The taxes - The discount - And the total cost . - mechanic's name - list of the accomplished services with each service price, total price, Taxes, and Discount
3a	If the code is invalid the use case ends, and an error message is communicated.
7a	If the payment method is invalid the use case ends, and an error message is communicated.

The classes :

1. Customer :

   Attributes :
   - firstName:string
   - lastName:staring
   - phoneNumber:string
   - customerEID:string
   - Email:string

2. vehicle :

   Attributes:
   - make:string
   - model:string
   - Year:string
   - color:Enum
   - ID:string

3. Service:

   Attributes:

- name:string
- description:string
- duration : string
- price:float

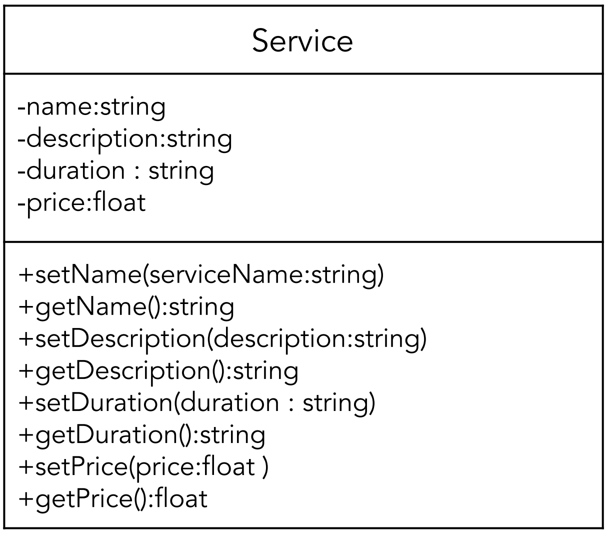
4. Receipt :

   Attributes:

   - customer :Customer
   - mechanic:string
   - vechicle:Vehicle
   - services:List[Services]
   - date:date
   - taxes:float
   - discount:float

Customer
-firstName:string -lastName:staring -phoneNumber:string -customerEID:string -email:string
+setFirstName(firstName:String) +getFirstName():string +setLastName(lastName:string) +getLastName():string +setPhoneNumber():string +getPhoneNumber(phoneNumber:string) +getID():string +setEmail(email:string) +getEmail():string

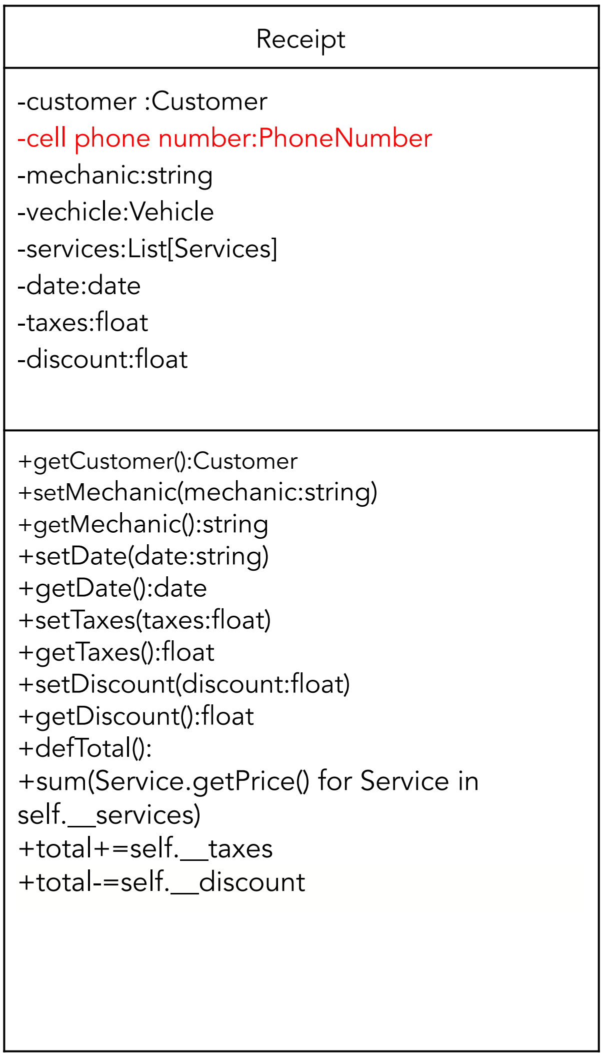
Vehicle
-make:string -model:string -year:string -color:Enum -ID:string
+setMake(make:string) +getMake():string +setModel(model:string) +getModel():string +setYear(year:string) +getYear()string +setColor(color:Color) +getColor():ENUM +getID():string



Service

-name:string
-description:string
-duration : string
-price:float

+setName(serviceName:string)
+getName():string
+setDescription(description:string)
+getDescription():string
+setDuration(duration : string)
+getDuration():string
+setPrice(price:float)
+getPrice():float



Receipt

-customer :Customer
-cell phone number:PhoneNumber
-mechanic:string
-vechicle:Vehicle
-services:List[Services]
-date:date
-taxes:float
-discount:float

+getCustomer():Customer
+setMechanic(mechanic:string)
+getMechanic():string
+setDate(date:string)
+getDate():date
+setTaxes(taxes:float)
+getTaxes():float
+setDiscount(discount:float)
+getDiscount():float
+defTotal():
+sum(Service.getPrice() for Service in self.__services)
+total+=self.__taxes
+total-=self.__discount