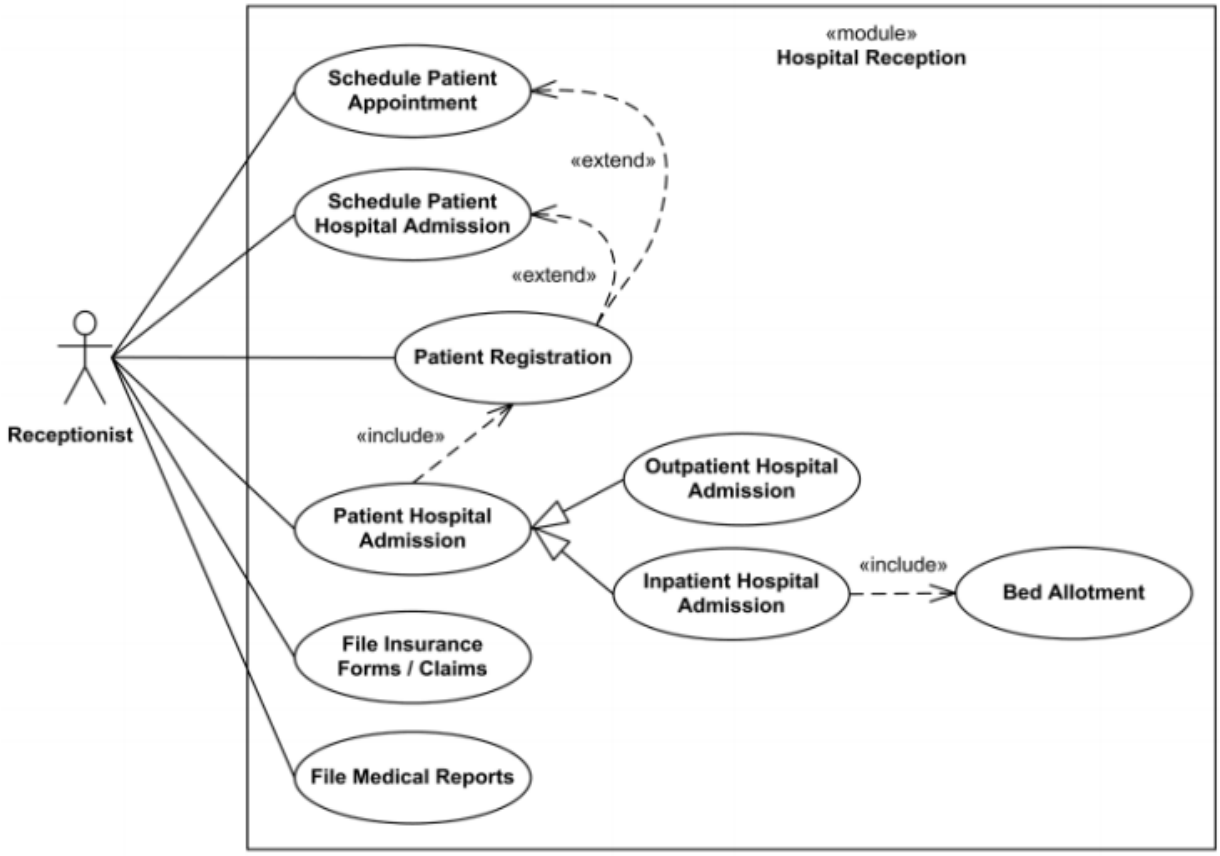
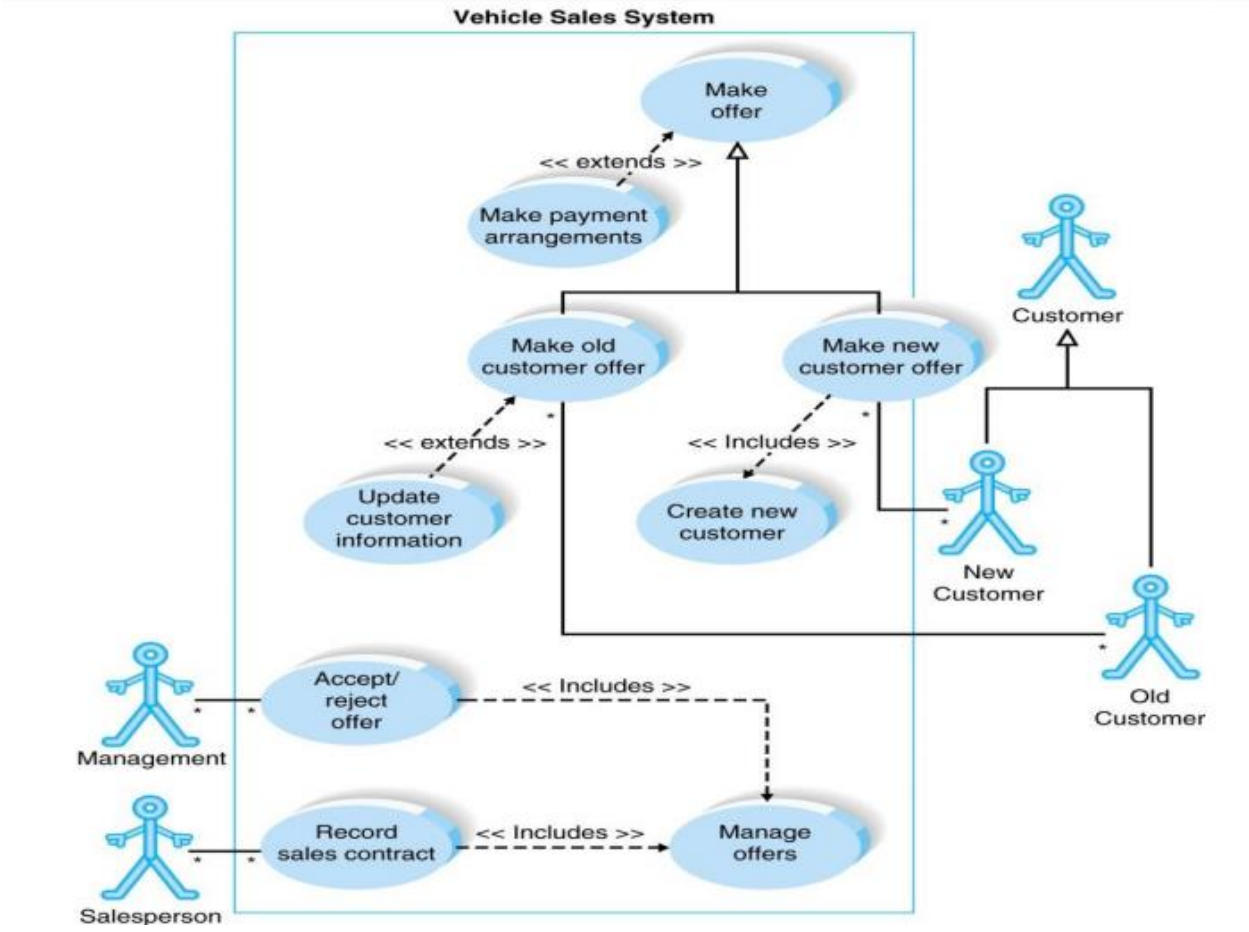
Use-case Diagram:

Draw a use case diagram for the hospital reception system. In this system, receptionist can schedule patient appointment and patient hospital admission after the patient registration. Both types of patients i.e. outpatient and inpatient can be admitted in the hospital. Receptionist also checks the insurance and claim forms and put them in file. Patient medical report is also filed by the receptionist.

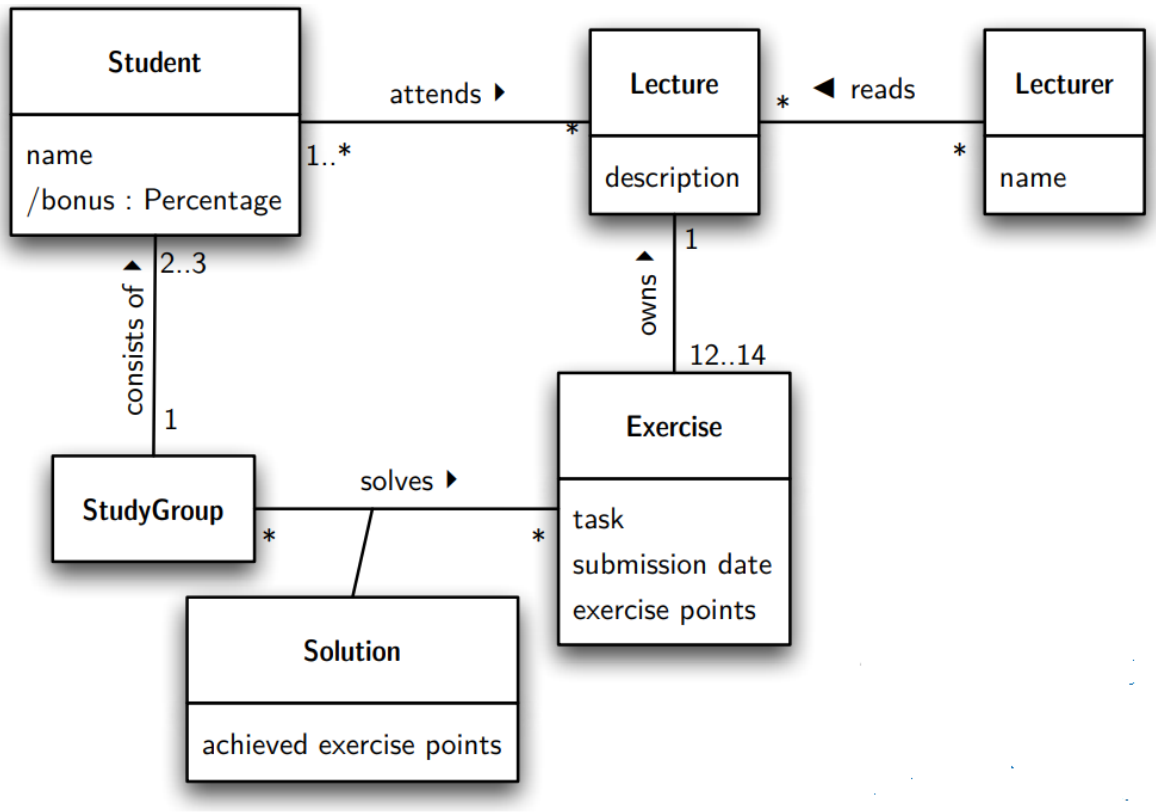


Draw a use case diagram for the vehicle sales system. Customer makes offer for the vehicle. Customer can be new customer or old customer. New and old customer can make their own offers. For every customer they have to get registered. System can update the existing customer information as well. Customer make payment if his/ her offer is accepted. Management has right to accept or reject the offer by managing the offer. Sales person records the sales contract of the accepted offer.



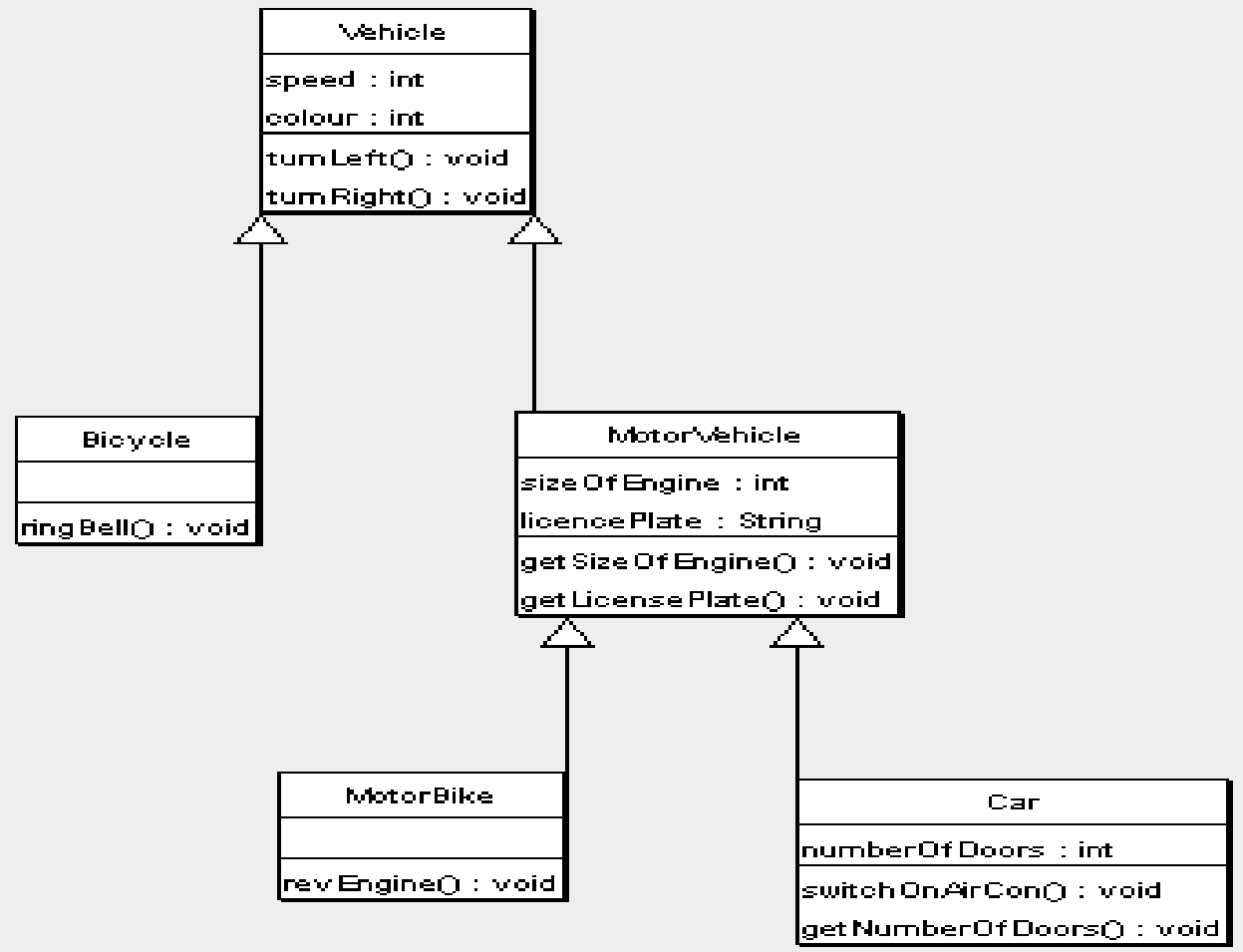
Domain Model

During a semester a lecturer reads one or more lectures. Sometimes the lecturer is on leave to focus on doing research, in this case (s)he does not give a lecture. A student usually attends one or more lectures, unless (s)he has something better to do. During the semester there will be several exercises which are meant to be solved by small study groups. Each student is assigned to one particular study group for the whole semester. A study group consists of two to three students. After submission of a solution by a study group it is graded by a tutor.

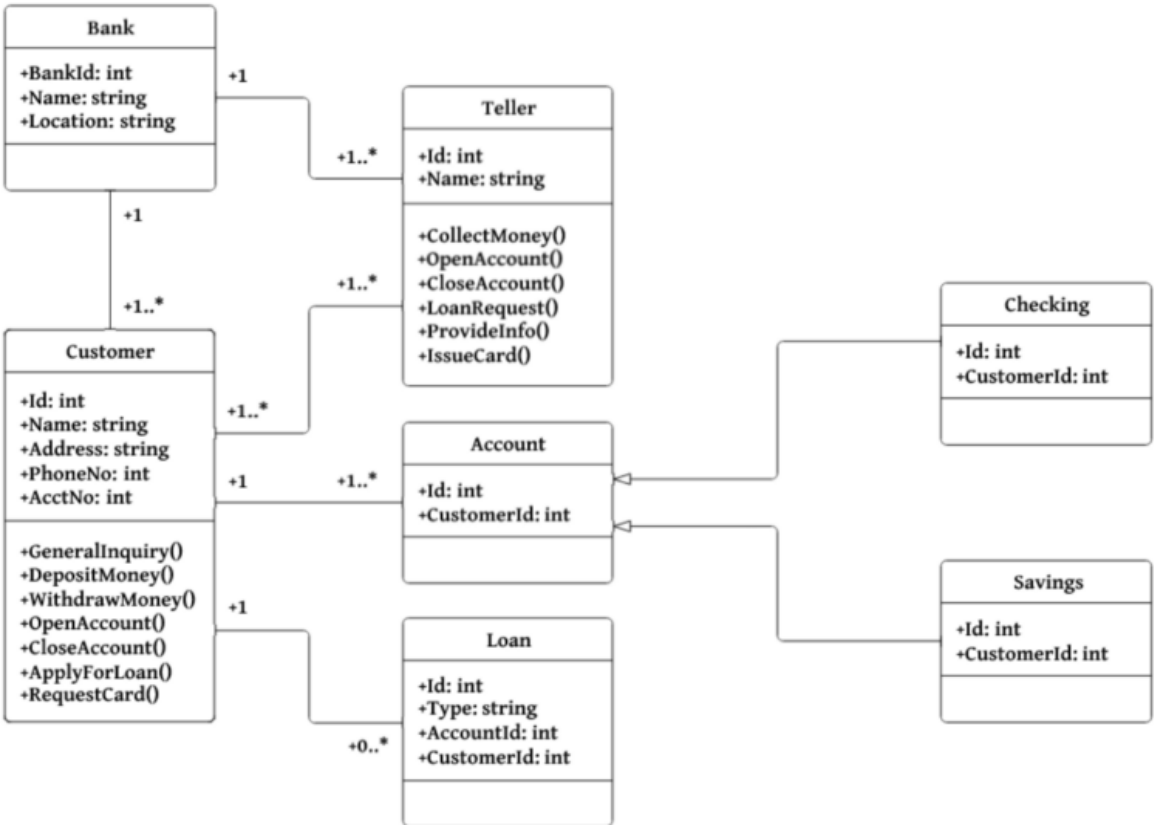


Class Diagram:

We have to develop an application that model different kinds of vehicles such as bicycles, motor bike and cars. All Vehicles have some common attributes (speed and colour) and common behavior (turnLeft, turnRight). Bicycle and MotorVehicle are both kinds of Vehicle. MotorVehicles have engines and license plates. MotorVehicles includes two types i.e. MotorBike and Car.



We have to develop a banking system application which provides many services to the customers like opening and closing accounts, balance enquiry, deposit money, cash withdrawl, and taking cards. Customer can open two types of accounts i.e. saving and current account. Bank also has an ATM machine which provides the services related to balance. Customer can take loan from the bank against his/her account. One customer can take only one loan at a time.

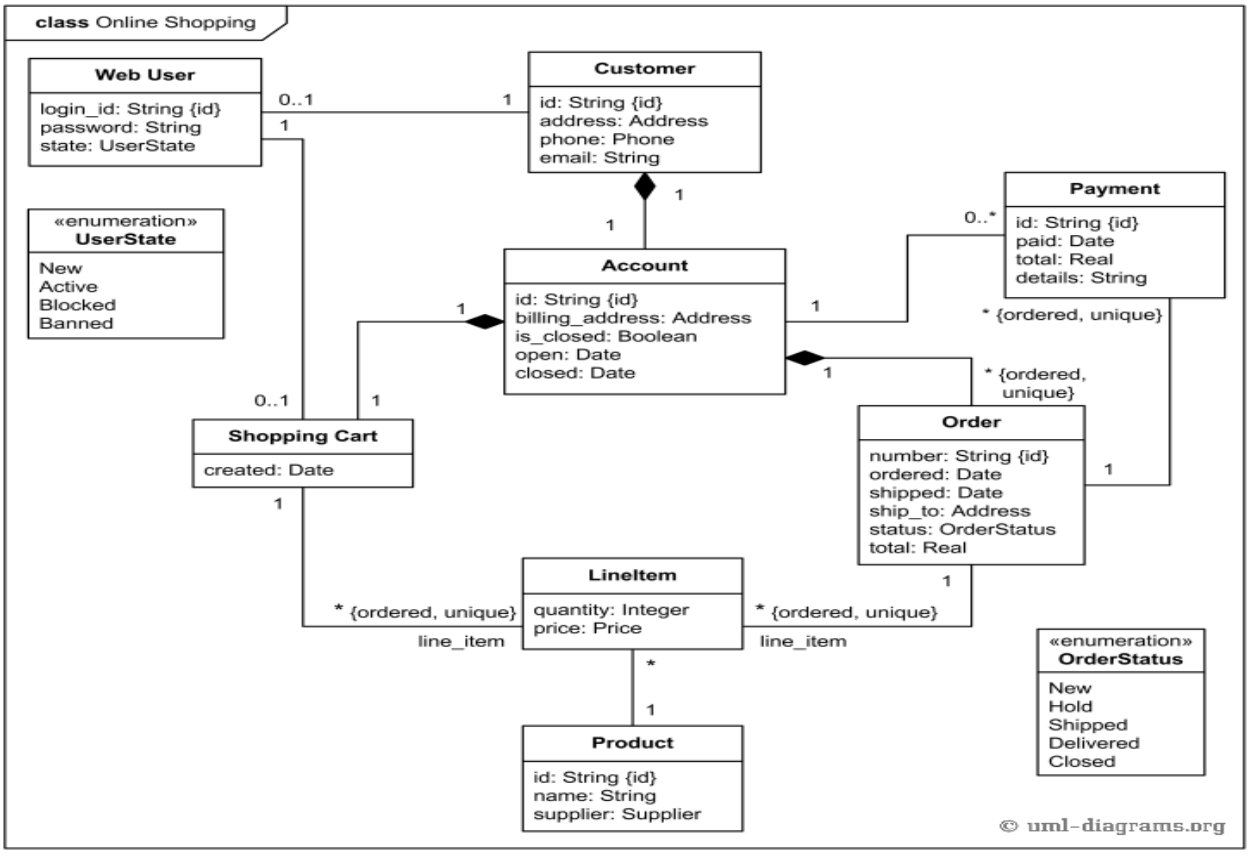


Draw the UML **class diagram** showing the domain model for online shopping. The purpose of the diagram is to introduce some common terms, "dictionary" for online shopping - Customer, Web User, Account, Shopping Cart, Product, Order, Payment, etc. and relationships between. It could be used as a common ground between business analysts and software developers.

Each customer has unique id and is linked to exactly one **account**. Account owns shopping cart and orders. Customer could register as a web user to be able to buy items online. Customer is not required to be a web user because purchases could also be made by phone or by ordering from catalogues. Web user has login name which also serves as unique id. Web user could be in several states - new, active, temporary blocked, or banned, and be linked to a **shopping cart**. Shopping cart belongs to account.

Account owns customer orders. Customer may have no orders. Customer orders are sorted and unique. Each order could refer to several **payments**, possibly none. Every payment has unique id and is related to exactly one account.

Each order has current order status (new, hold, shipped, delivered, closed). Both order and shopping cart have **line items** linked to a specific product. Each line item is related to exactly one product. A product could be associated to many line items or no item at all.


class Online Shopping
Web User
login_id: String {id}
password: String
state: UserState
0..1
1
1
Customer
id: String {id}
address: Address
phone: Phone
email: String
1
1
«enumeration»
UserState
New
Active
Blocked
Banned
Account
id: String {id}
billing_address: Address
is_closed: Boolean
open: Date
closed: Date
1
1
0..*
Payment
id: String {id}
paid: Date
total: Real
details: String
* {ordered, unique}
1
* {ordered,
unique}
0..1
1
Shopping Cart
created: Date
1
Order
number: String {id}
ordered: Date
shipped: Date
ship_to: Address
status: OrderStatus
total: Real
1
1
* {ordered, unique}
line_item
LineItem
quantity: Integer
price: Price
* {ordered, unique}
line_item
*
1
Product
id: String {id}
name: String
supplier: Supplier
«enumeration»
OrderStatus
New
Hold
Shipped
Delivered
Closed
© uml-diagrams.org