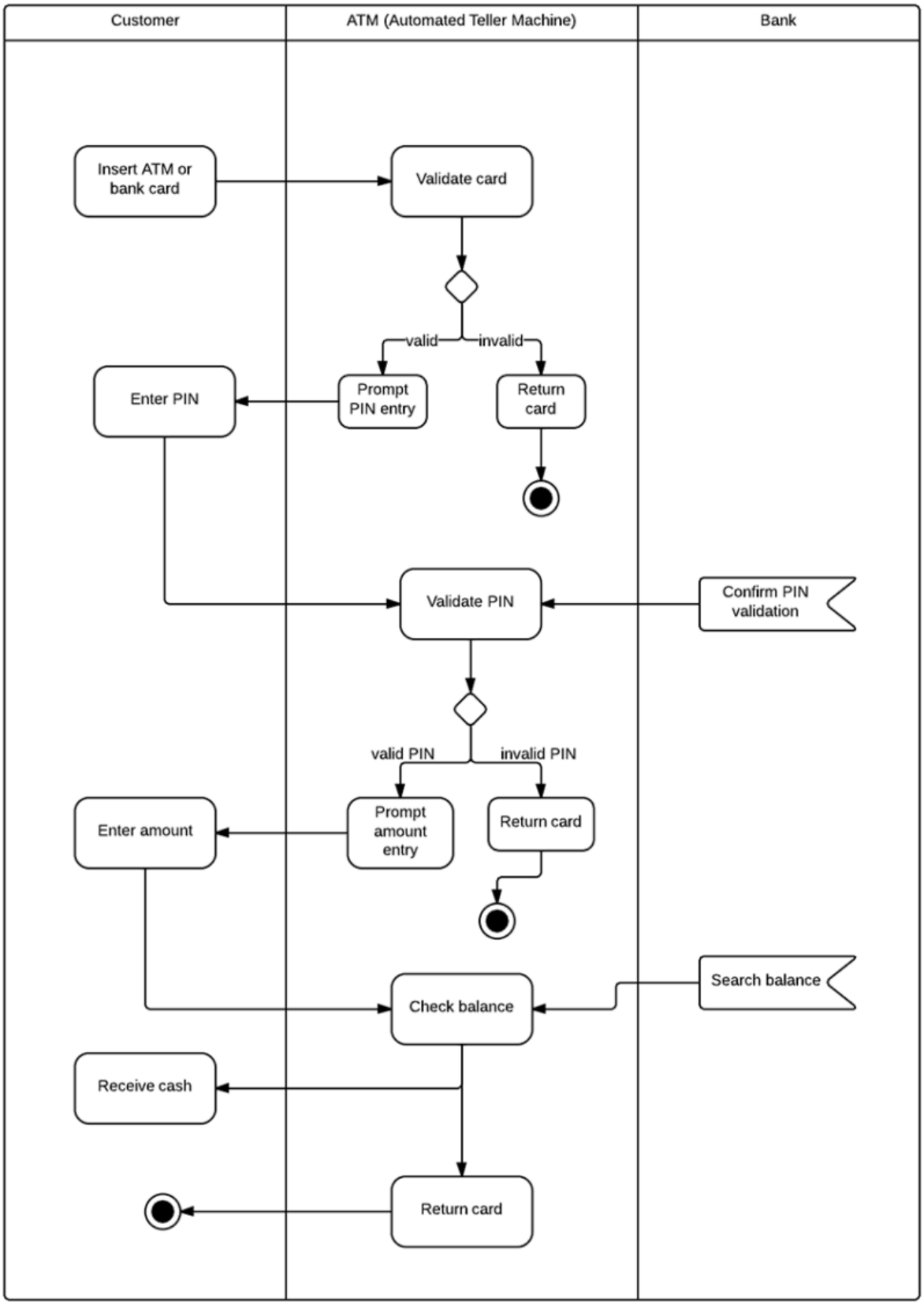

Customer
ATM (Automated Teller Machine)
Bank
Insert ATM or bank card
Validate card
valid
invalid
Enter PIN
Prompt PIN entry
Return card
Validate PIN
Confirm PIN validation
valid PIN
invalid PIN
Enter amount
Prompt amount entry
Return card
Search balance
Check balance
Receive cash
Return card

Above and Below is Q4.-Activity diagram for ATM System

3. The following is the use case description of using ATM machine to withdraw money from a bank account. Draw a UML activity diagram according to the description.

Name of use case: Withdraw money from bank account

Actors: *Customer*: A customer is client of the bank who has an account. A customer can deposit and withdraw money to his/her account by using an ATM machine, and make queries of the balance on his/her account.

Entry condition:

a. The customer has a cash card of the account, which records the account number.
b. The ATM machine is in the state of 'Ready to Serve'.

Exit condition:

a. The customer has obtained the money as he/she required and the balance of the account is updated.

Flow of events:

1. The customer inserts the card into the ATM machine;
2. The ATM machine reads the account number recorded on the card;
3. The ATM machine request the bank's service computer to check card validity;
4. The bank's server confirms the validity;
5. The ATM machine prompt the client to input PIN;
6. The customer enters PIN;
7. The ATM machine sends the PIN to the bank's server to check if the PIN is correct.
8. The bank's server confirms the PIN is correct;
9. The ATM machine prompts the customer to select a service from 'Withdraw cash', 'Request Statement' and 'Balance Query'.
10. The customer selects 'Withdraw cash'.
11. The ATM machine send a message to the bank's server to request send the amount of money available to withdraw;
12. The Sever sends the ATM machine the amount available to withdraw.
13. The ATM machine displays the amount available and prompts the customer to enter the amount required.
14. If the amount is less than or equals to the amount available, the ATM machine returns the card.
15. The customer takes the card.
16. The ATM machine dispatches the cash.
17. The customer takes the cash.
18. The ATM machine sends a message to the bank's server to report the completion of the transaction.
19. The ATM returns the state 'Ready to Serve' the next customer.

Exceptions and alternative flow of events

a. The card is invalid:
4.1. (1) The bank's server replies the ATM with a message that the card is invalid.

(2) The ATM display a message on the screen to inform the customer that the card is invalid.

(3) The ATM machine returns the card to the customer.

(4) The system returns to the state of 'Ready to Serve' to the next customer.

b. *The PIN number that the customer entered is incorrect:*

8.1. (1) The banks server replies that the PIN is incorrect;

(2) The ATM machine displays a message on the screen to inform that the PIN is incorrect;

(3) Back to step 5.

c. The required amount is greater than the amount available:

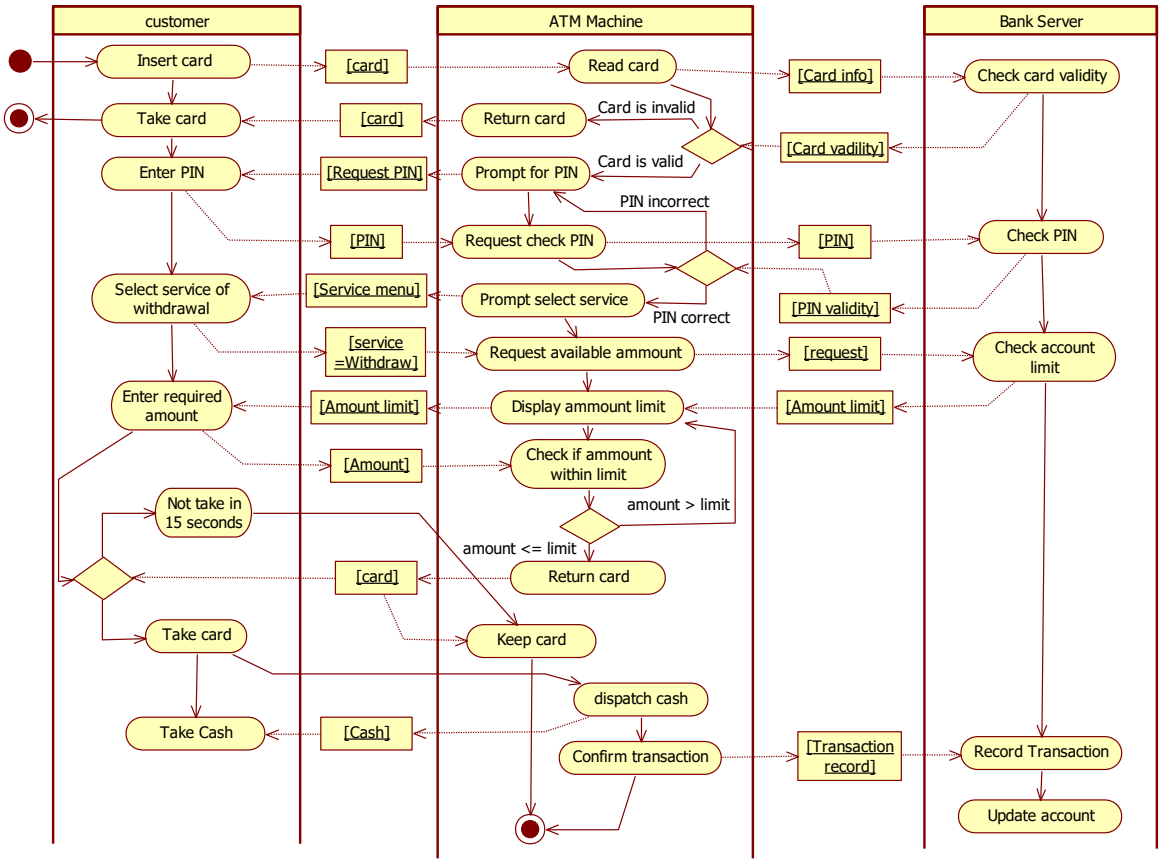
14.1. (1) The system displays a message that the amount is not available;

(2) Back to step 13.

d. The customer does not take card for 15 second:

15.1 (1) The ATM machine takes the card display a message to the customer that he/she can get the card back from his/her bank branch.

(2) The system returns to the state of 'Ready to Serve' the next customer.



SAMPLE ANSWER:

1. The following is a narrative description of the business process of organisation of conferences with regards to the submitting, reviewing and accepting papers.

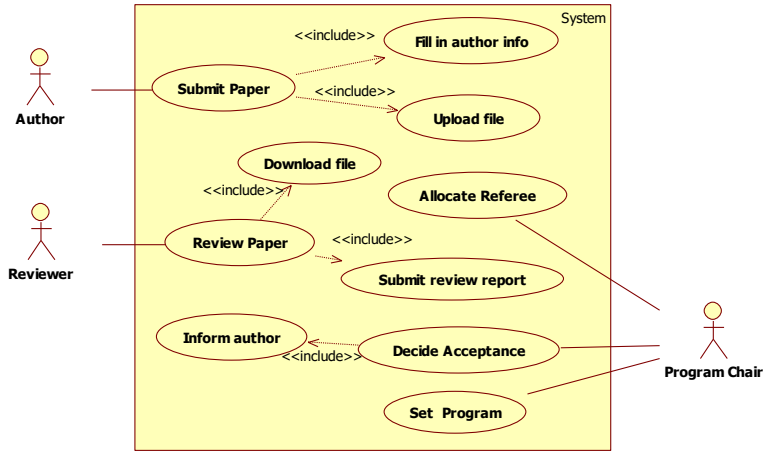
Q5. ACTIVITY DIAGRAM FOR ONLINE PAPER SUBMISSION SYSTEM

The author completes an online form that requests the user to input author name, correspondence address, email and, title of paper. The system validates this data and, if correct, asks the author to submit the paper. The author then browses to find the correct paper on their system and submits it. Once received and stored, the system returns to the author a reference number for the paper. Authors may submit as many papers as they like to be considered for acceptance to the conference up until the deadline date for submissions. Papers are allocated to referees for assessment. They review each paper and submit to the system their decision. Once the programme organiser has agreed the decisions authors are informed by email. Accepted papers are then schedule to be delivered at a conference. This involves allocating a date, time and place for the presentation of the paper.

1.1 Analyse the above text and then draw a use case diagram using an UML modelling tool such as StarUML.

SAMPLE ANSWER:

1.2. Derive an activity diagram from the narrative text by following the steps outlined below.



(1) Translate the above narrative text into an activity list. An activity list consists of a sequences of short sentences in the form of

name + verb + noun [*+ condition*],

where the *condition* is optional. Such sentences are called activities. For example, the following is an activity.

'*the author completes an online form*'.

(2) Transform each activity in the list into an activity/action node of the activity diagram;
(3) Identify the actors involved in the business process, and associate the actors to each action/activity node;
(4) Identify the sequential execution order, parallelism/concurrency and synchronisation as well between the actions/activities, and draw the control flows between the action/activity nodes, add synchronisation bar as necessary;
(5) Draw a swimlane for each actor and place the action/activity nodes in appropriate swimlane;
(6) Identify the objects passed between the activities, and add object nodes and object flows into the diagram.

SAMPLE ANSWER:

(1) The activity list is given below.

1) The system **requests the user to input** author name, correspondence address, email and, title of paper if the date is before the deadline.
2) The author **completes an online form** that contains the data.
3) The system **validates this data**.
4) The system **asks the author to submit** the paper, if the data is correct.
5) The author **browses their system**.
6) The author **find the correct paper** on their system.
7) The author **submits the paper**.
8) The system **receives the paper**.
9) The system **stores the paper**.
10) The system **assigns a reference number** for the paper.
11) The system **returns the reference number** to the author.
12) The author goes back to activity 4) if they want to submit another paper.
13) System **allocates papers to referees** for assessment.
14) Referee **reviews each paper**.
15) Referee **submits their assessments** to the system.
16) The programme organiser **make decisions** on each paper's acceptance.
17) The system **emails the author** of each paper of the decision.
18) The system **creates a schedule** for delivering accepted papers at a conference by allocating a date, time and place for the presentation of each paper.

(2) The action/activity nodes are marked on the activity list as bold font.

(3) The actors are the name part of the activities.

(4) In this exercise, the actions are all sequentially ordered as in the activity list. The parallelism and concurrency only occur in the form of different referees can perform their reviews and assessments in parallel.

(5) The actors are:

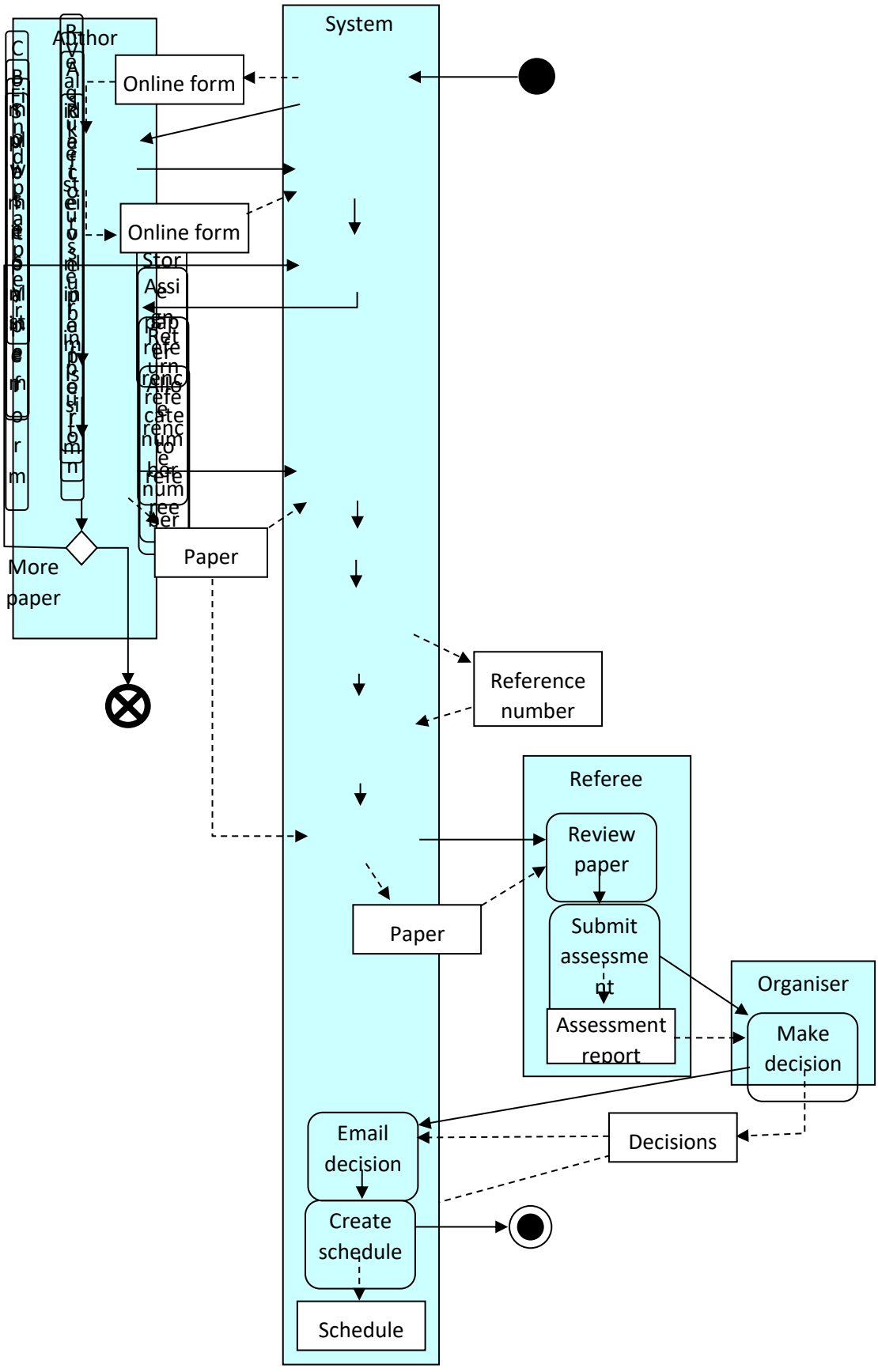
(a) Author; (b) The system; (c) Referee; (d) Conference programme organiser;

(6) The objects involved in this process are:

(a) online form; (b) paper; (c) paper's reference number; (d) assessment report;

(e) decision; (f) conference schudel.

The derived activity diagram is given below.


Author
System
Online form
Online form
Paper
More paper
Reference number
Referee
Review paper
Submit assessment
Assessment report
Paper
Organiser
Make decision
Decisions
Email decision
Create schedule
Schedule

1.3. Review the activity diagram to identify information missing in the narrative text.

SAMPLE ANSWER:

There are a number problems that can be found and improved in the examination of the activity diagram.

First, what if a validation action fails? Such as if the online form is not valid, what action(s) should be taken? The diagram is incomplete on this issue.

Second, if an action/activity involves two participants, the *duality principle* of task analysis developed in HCI theory requires each participant to have a corresponding action/activity. For example, the submission of an online form to the system is an action of the author. It requires the system to take the action of receiving an online form. In the narrative text, this duality is often taken for granted, thus usually not given explicitly. Therefore, it is not represented in the activity diagram as derived by following the above steps. Consequently, the diagram is incomplete. Therefore, more action(s)/activities should be added into the diagram. These problems are because that the diagram is directly translated from the narrative text.

Q6. The following is a use case description of the examination paper preparation support system. Draw a UML activity diagram according to the description.

Use case name: submit question

Participant: lecturer

Entry conditions:

1. The question is ready and stored in a file
2. The lecturer is assigned to the module

Exit conditions:

1. The file is uploaded to the system
2. The module leader is notified of the availability of the question
3. The event is logged by the system

Flow of Events:

1. The lecturer logs into the system by entering his/her username and password;
2. The system checks the username and password;
3. The system displays the list of modules of which he/she is the lecturer, module leader and/or internal examiner;
4. The lecturer selects a module and his/her role in the module as a lecturer;
5. The system prompts the user to enter the file name and location on his/her computer, and additional information if any;
6. The lecturer enters file name and location, and types in the additional information;
7. The lecturer submits the questions and the file is uploaded to the system;
8. The system saves the file;
9. The system confirms the success of uploading the file.
10. The system notifies the module leader of the submission of the questions.

Exceptional conditions and alternative flow of events:

When the username and password is not correct:

3.1: display error message, go back to step 1;

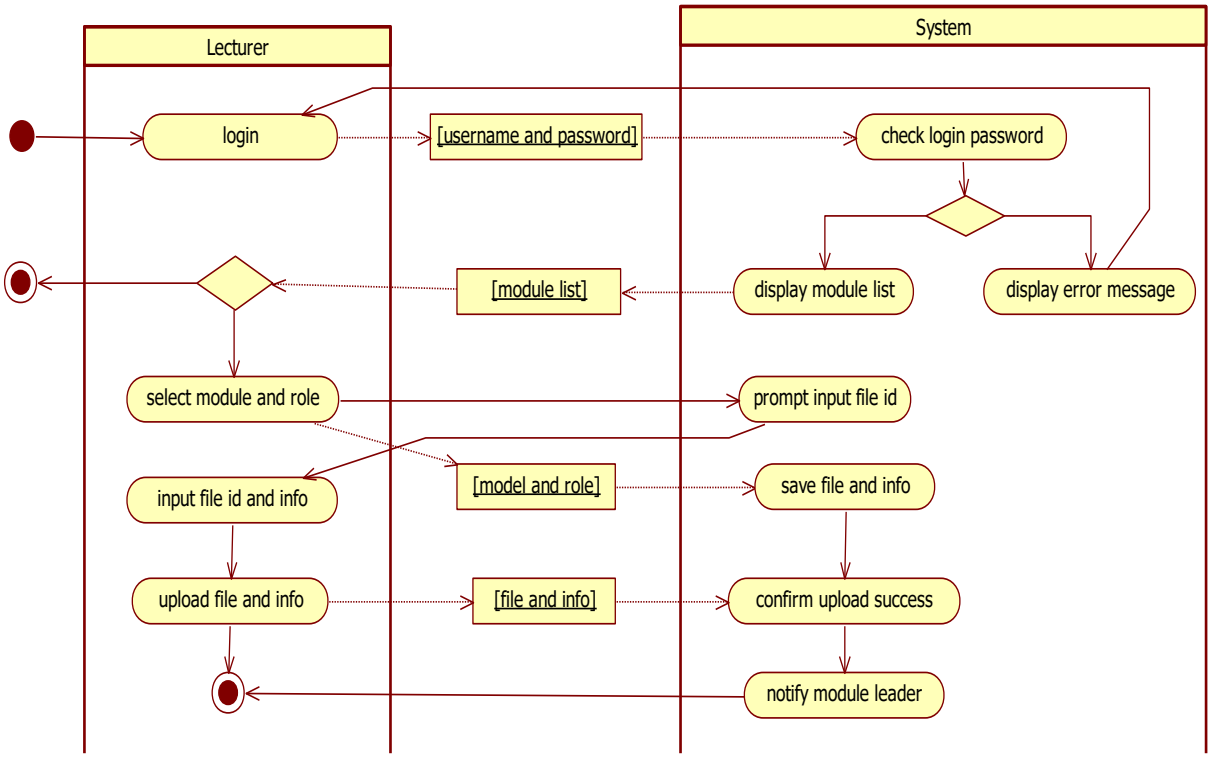
When the lecturer is not listed on the module:

4.1: quit the system;

Special requirements:

1. The file should be encrypted when transmitted from lecturer's computer to the server
2. The notification of success in uploading the file should be within 20 seconds
3. The event should be recorded in a log file to contain the following information:
   a) name of the lecturer,
   b) date and time of the event,
   c) the name of the event (upload exam question),
   d) the file on the server that stores the questions.

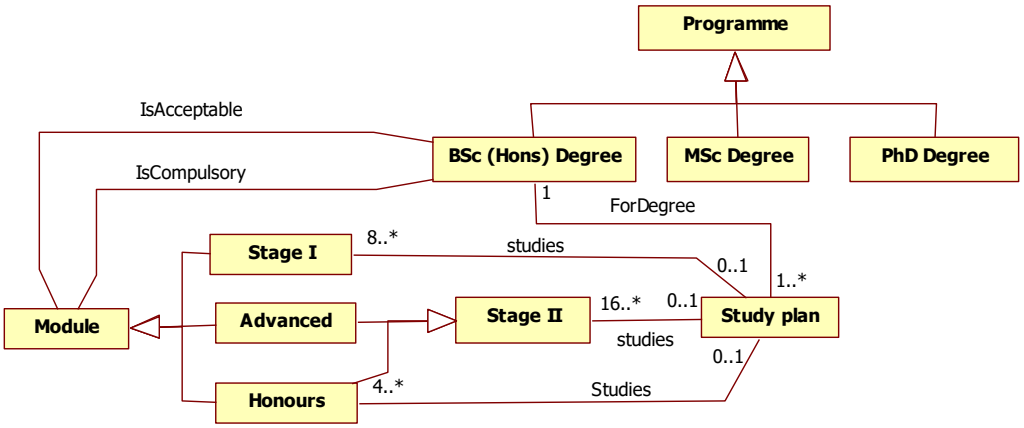
SAMPLE ANSWER:



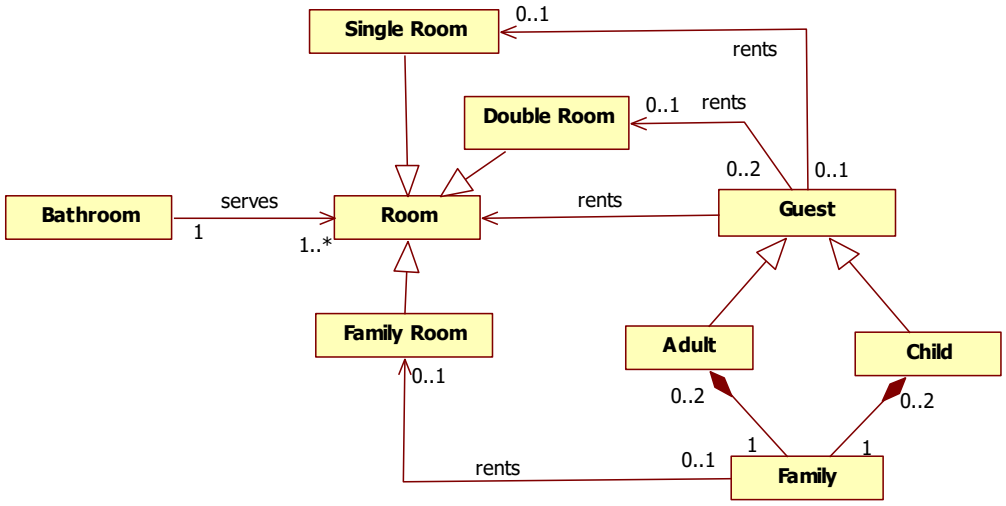
4. The following are description of systems. Draw a UML class diagram to represent the structural model for each of them.

4.1. *A university offers a number of degree programmes, which are classified into BSc (Hons) degree programme, MSc degree program and PhD degree programme. To teach students in various programmes, the university runs a number of course modules. A particular module could be acceptable to a program, or compulsory to a program, or not acceptable at all to the programme. Each BSc (Hons) degree programme contains a number of modules as acceptable or compulsory, which are classified into stage I modules, advanced modules and honours modules. For a student who studies a BSc (Hons) degree programme, in order to obtain the degree he/she must complete a study plan that consists of at least 8 stage I modules, 16 advanced/honours modules and 4 honours modules that are acceptable to the programme.*

SAMPLE ANSWER:



4.2. A hotel has a number of rooms that can be rent by guests. There are also a number of bathrooms, which are either connected to a specific room or are used to service multiple rooms on the floor. The rooms are classified into three types: single rooms, double rooms and family rooms. Each single room can only be rent to at most one guest. Each double room can be rent to at most two guests. Each family room can be rent to a family of up to two adults and two children.



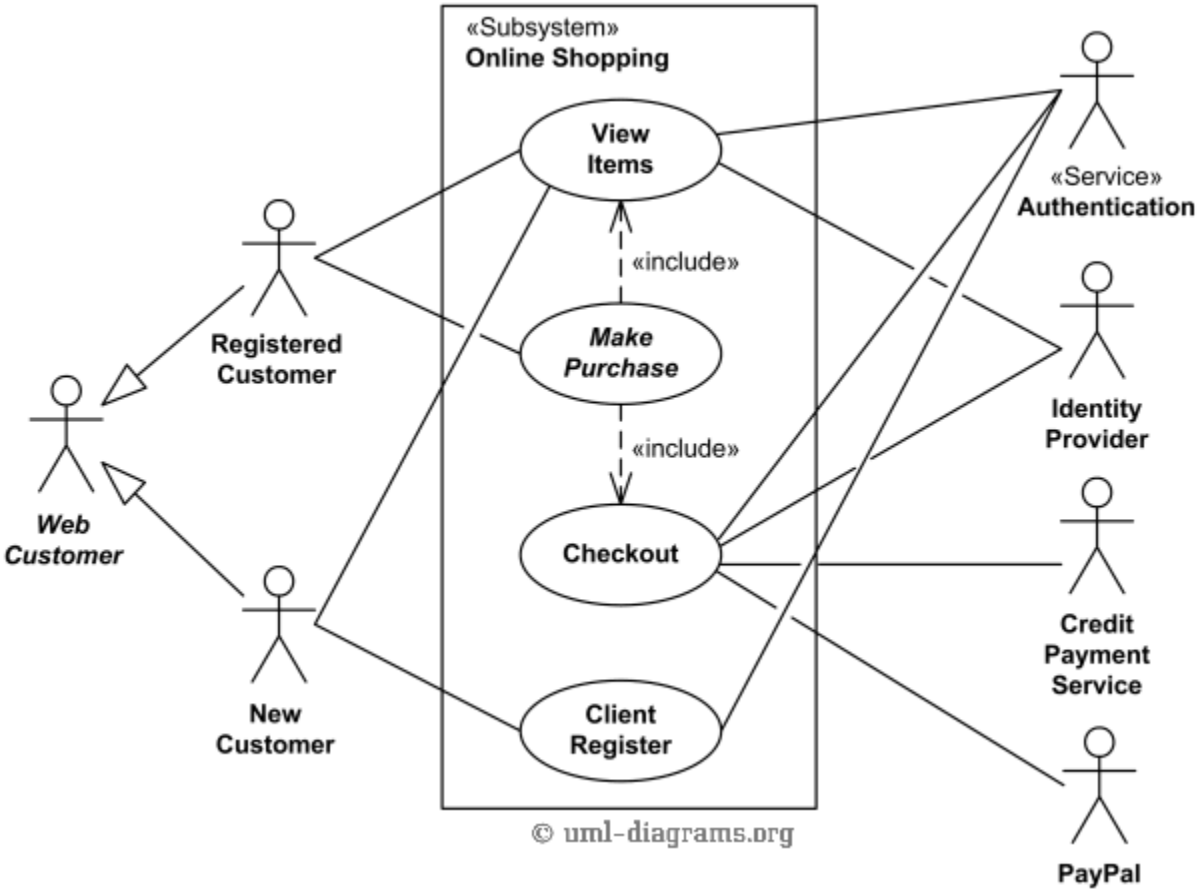
SAMPLE ANSWER:

Q8. Online Shopping

UML Use Case Diagram Example

Web Customer actor uses some web site to make purchases online. Top level use cases are **View Items**, **Make Purchase** and **Client Register**. View Items use case could be used by customer as top level use case if customer only wants to find and see some products. This use case could also be used as a part of Make Purchase use case. Client Register use case allows customer to register on the web site, for example to get some coupons or be invited to private sales. Note, that **Checkout** use case is included use case not available by itself - checkout is part of making purchase.

Except for the **Web Customer** actor there are several other actors which will be described below with detailed use cases.



Online shopping UML use case diagram example - top level use cases.

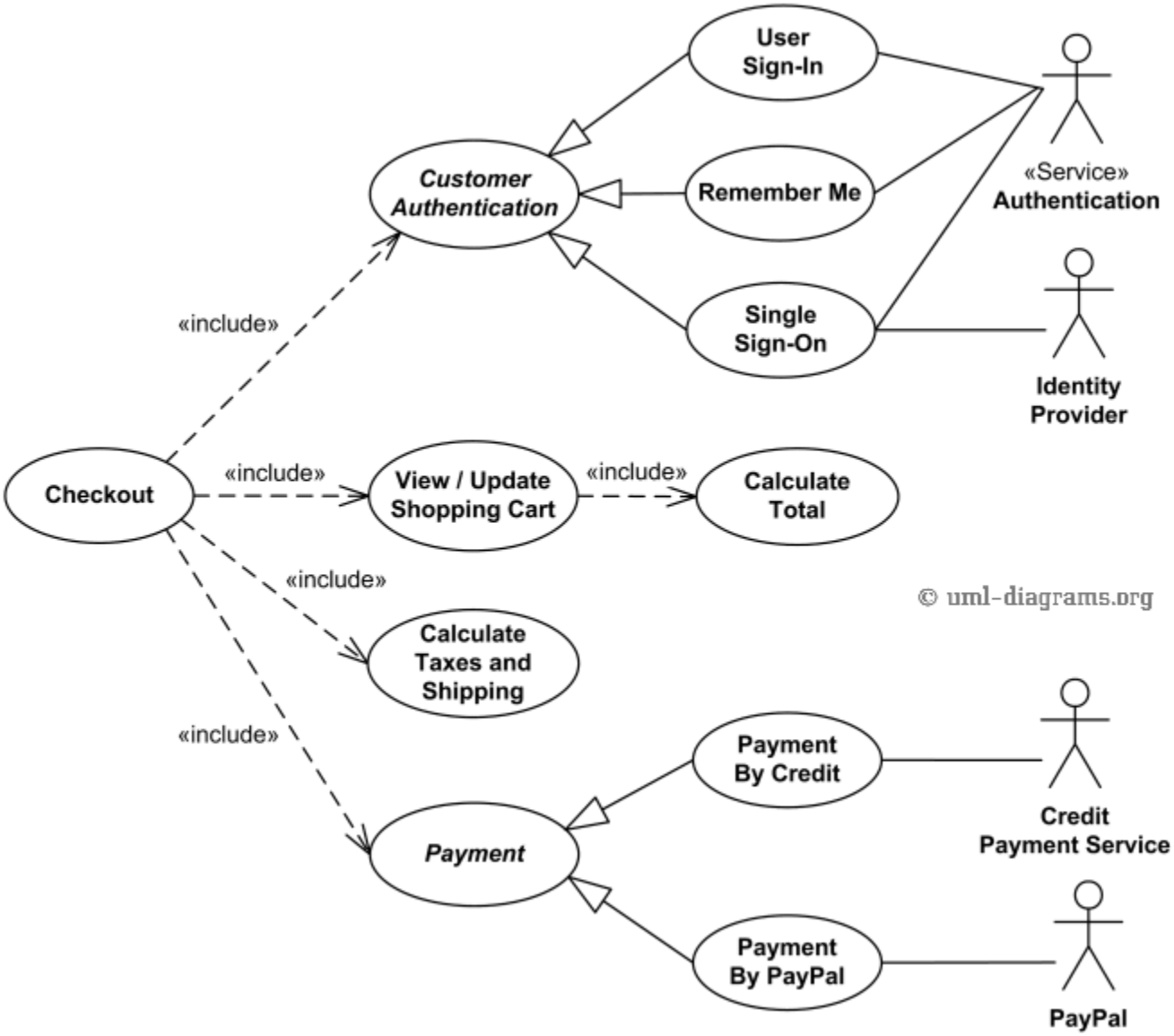
View Items use case is extended by several optional use cases - customer may search for items, browse catalog, view items recommended for him/her, add items to shopping cart or wish list. All these use cases are extending use cases because they provide some optional functions allowing customer to find item.

Customer Authentication use case is included in **View Recommended Items** and **Add to Wish List** because both require customer to be authenticated. At the same time, item could be added to the shopping cart without user authentication.

Online shopping UML use case diagram example - view items use case.

Checkout use case includes several required uses cases. Web customer should be authenticated. It could be done through user login page, user authentication cookie ("Remember me") or Single Sign-On (SSO). Web site authentication service is used in all these use cases, while SSO also requires participation of external identity provider.

Checkout use case also includes **Payment** use case which could be done either by using credit card and external credit payment service or with PayPal.



Online shopping UML use case diagram example - checkout, authentication and payment use cases.

Online Shopping - Credit Cards Processing

UML Use Case Diagram Example

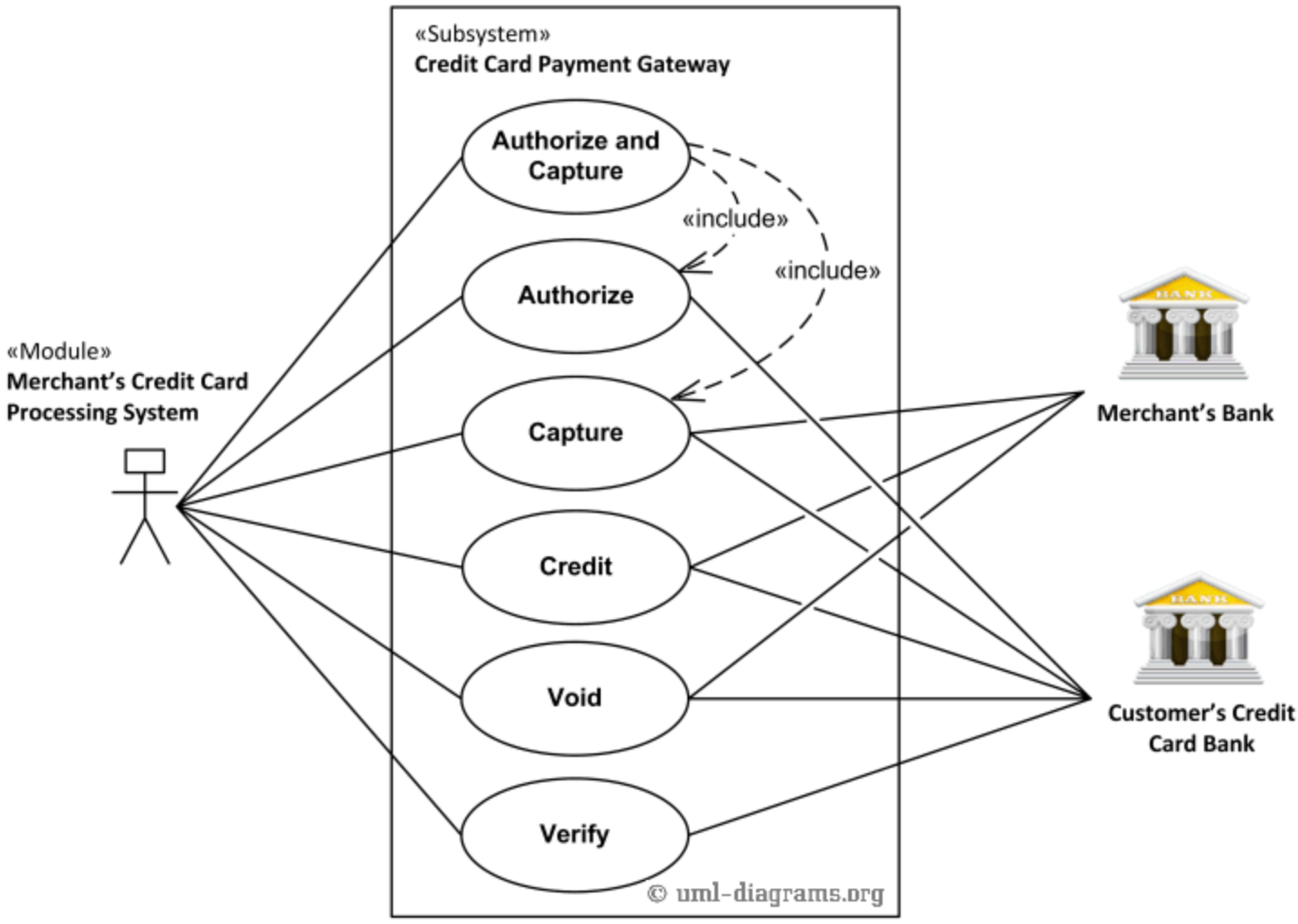
This UML use case diagram example shows some use cases for a system which processes credit cards.

Credit Card Processing System (aka Credit Card Payment Gateway) is a subject, i.e. system under design or consideration. Primary actor for the system is a **Merchant's Credit Card Processing System**. The merchant submits some credit card transaction request to the credit card payment gateway on behalf of a customer. Bank which issued customer's credit card is actor which could approve or reject the transaction. If transaction is approved, funds will be transferred to merchant's bank account.

Authorize and Capture use case is the most common type of credit card transaction. The requested amount of money should be first authorized by **Customer's Credit Card Bank**, and if approved, is further submitted for settlement. During the settlement funds approved for the credit card transaction are deposited into the **Merchant's Bank** account.

In some cases, only **authorization** is requested and the transaction will not be sent for settlement. In this case, usually if no further action is taken within some number of days, the authorization expires. Merchants can submit this request if they want to verify the availability of funds on the customer's credit card, if item is not currently in stock, or if merchant wants to review orders before shipping.

Capture (request to capture funds that were previously authorized) use case describes several scenarios when merchant needs to complete some previously authorized transaction - either submitted through the payment gateway or requested without using the system, e.g. using voice authorization.



UML use case diagram example for a credit cards processing system.

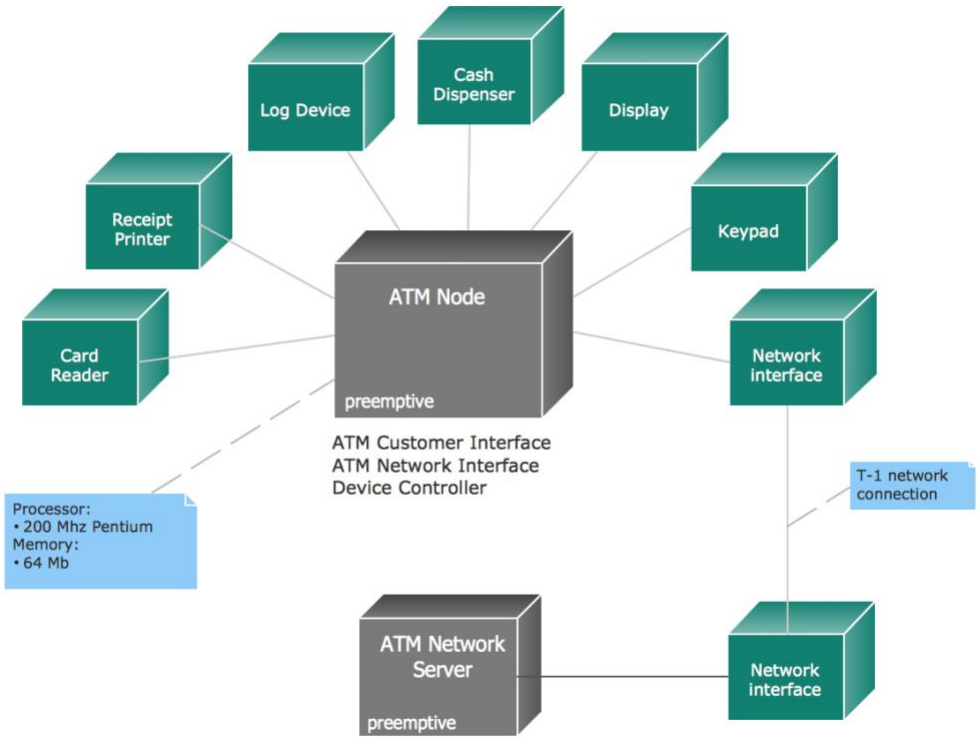
Credit use case describes situations when customer should receive a refund for a transaction that was either successfully processed and settled through the system or for some transaction that was not originally submitted through the payment gateway.

Void use case describes cases when it is needed to cancel one or several related transactions that were not yet settled. If possible, the transactions will not be sent for settlement. If the Void transaction fails, the original transaction is likely already settled.

Verify use case describes zero or small amount verification transactions which could also include verification of some client's data such as address.

You can find excellent resources, documentation, white papers, guides, etc. related to the credit card processing at Authorize.Net - Payment Gateway to Accept Online Payments.

DEPLOYMENT DIAGRAM FOR ATM SYSTEM



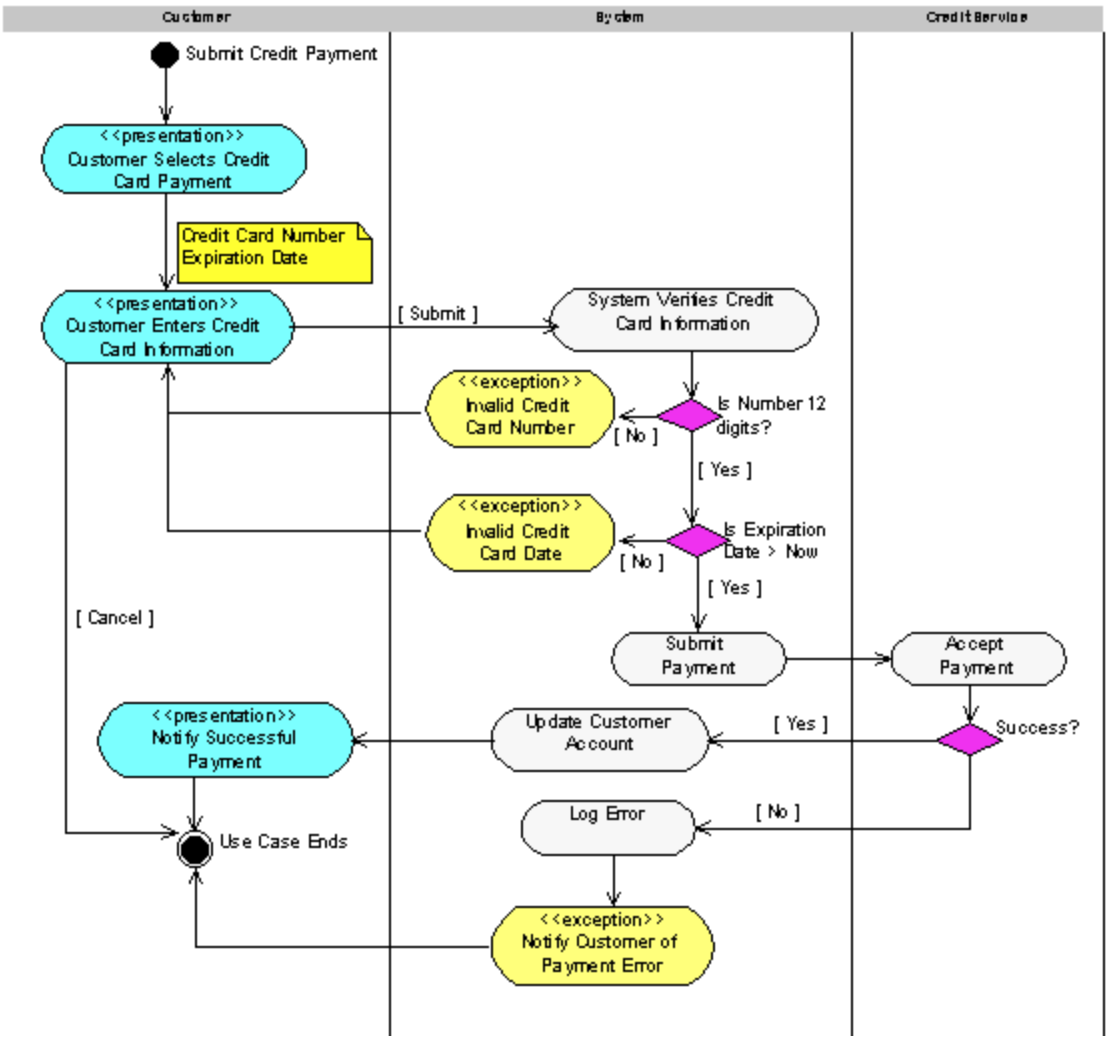
Use Case to accept credit Card Payment:

1. The customer then **enters** and submits her card details.

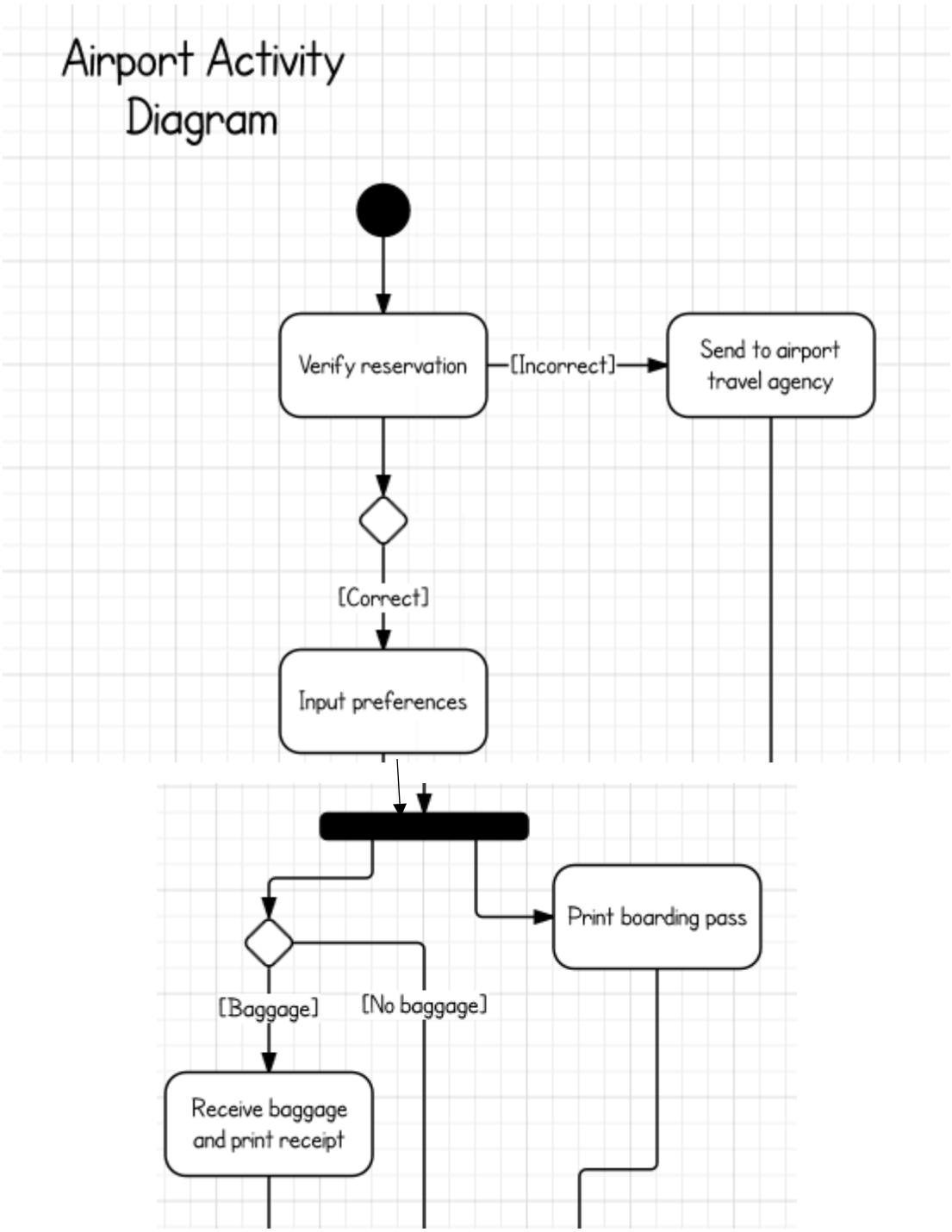
2. The system **validates** these values and either returns to the customer if there is an error or submits the payment to the Credit Card Service.

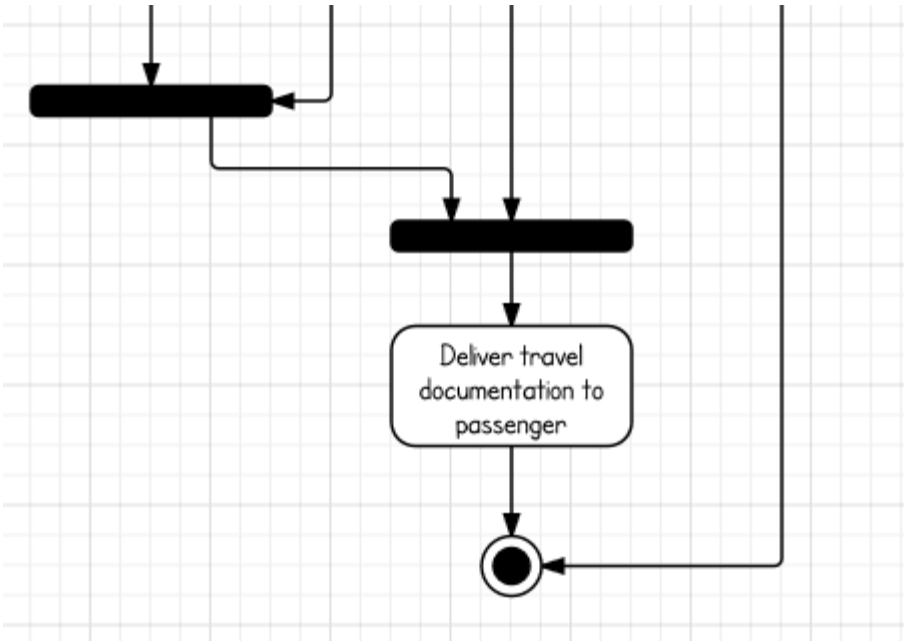
3. If the card payment is **accepted**, then the system notifies the customer of success. If not, then the **error** is logged, and the customer is notified of the failure (and perhaps directed to handle the payment some other way).

Q10. ACTIVITY DIAGRAM FOR CREDIT CARD PROCESSING

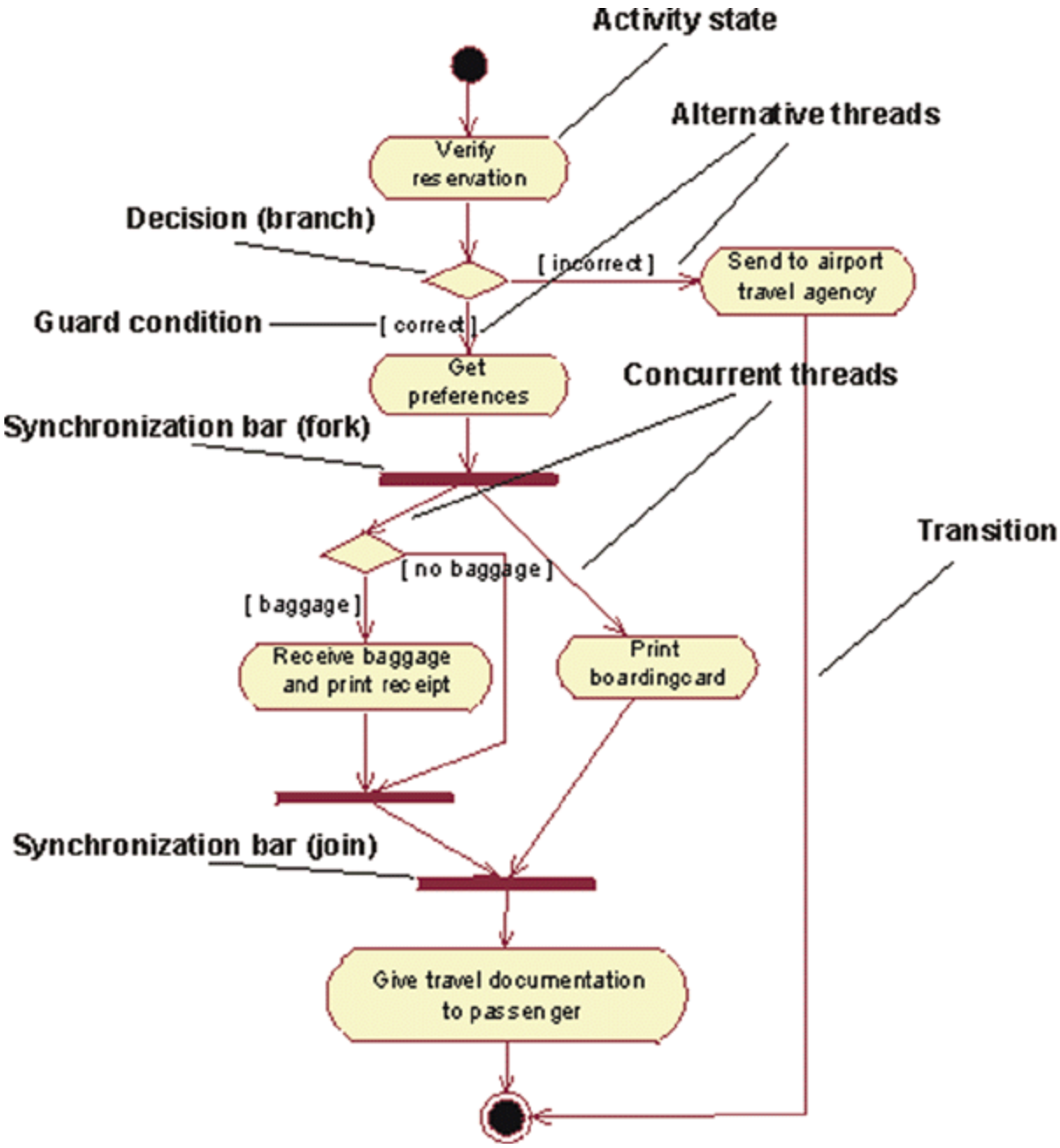


Q9. ACTIVITY DIAGRAM FOR AIRPORT CHECK-IN SECURITY SCREENING SYSTEM


Airport Activity Diagram
Verify reservation
[Incorrect]
Send to airport travel agency
[Correct]
Input preferences
Print boarding pass
[Baggage]
[No baggage]
Receive baggage and print receipt



OR



USE CASE DIAGRAM FOR AIRPORT CHECK-IN SECURITY SCREENING SYSTEM

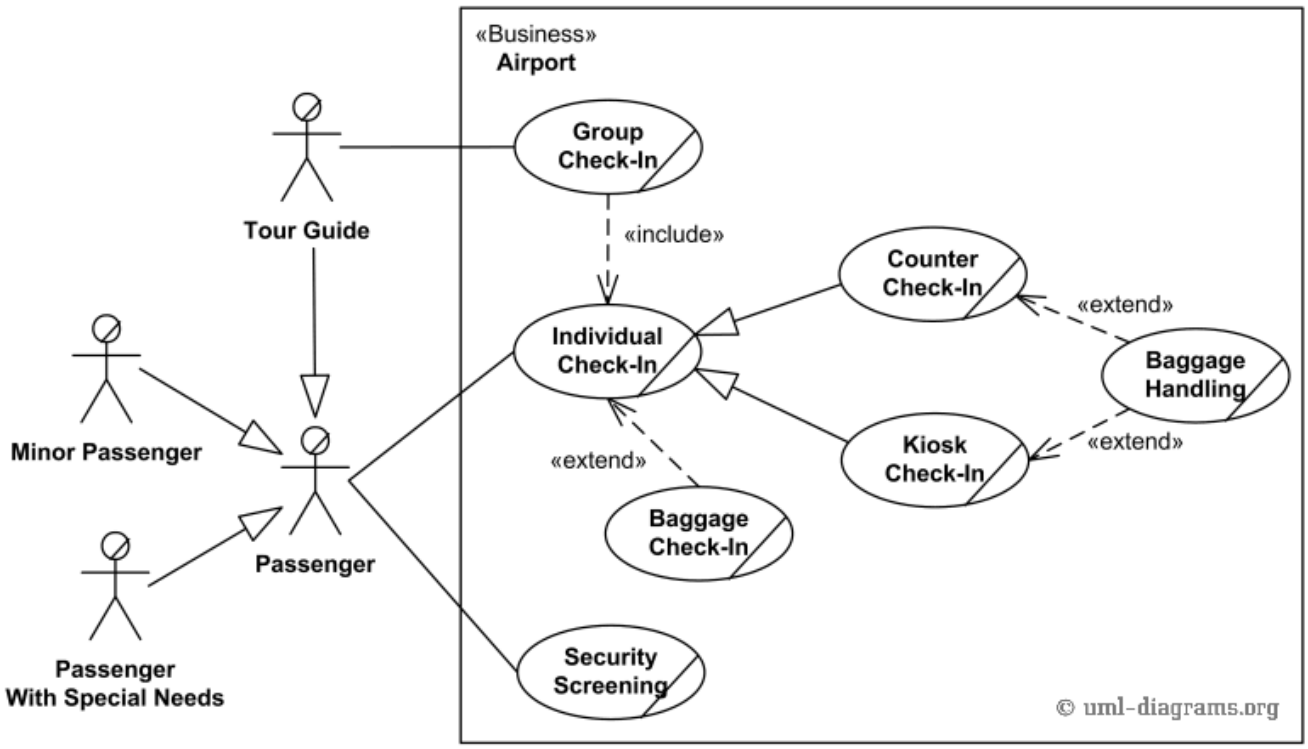
UML Use Case Diagram Example

This is an example of a business use case diagram which is usually created during **Business Modeling** and is rendered here in **Rational Unified Process** (RUP) notation.

Business actors are Passenger, Tour Guide, Minor (Child), Passenger with Special Needs (e.g. with disabilities), all playing **external roles** in relation to airport business.

Business use cases are Individual Check-In, Group Check-In (for groups of tourists), Security Screening, etc. - representing business functions or processes taking place in airport and serving the needs of passengers.

Business use cases Baggage Check-in and Baggage Handling extend Check-In use cases, because passenger might have no luggage, so baggage check-in and handling are optional.



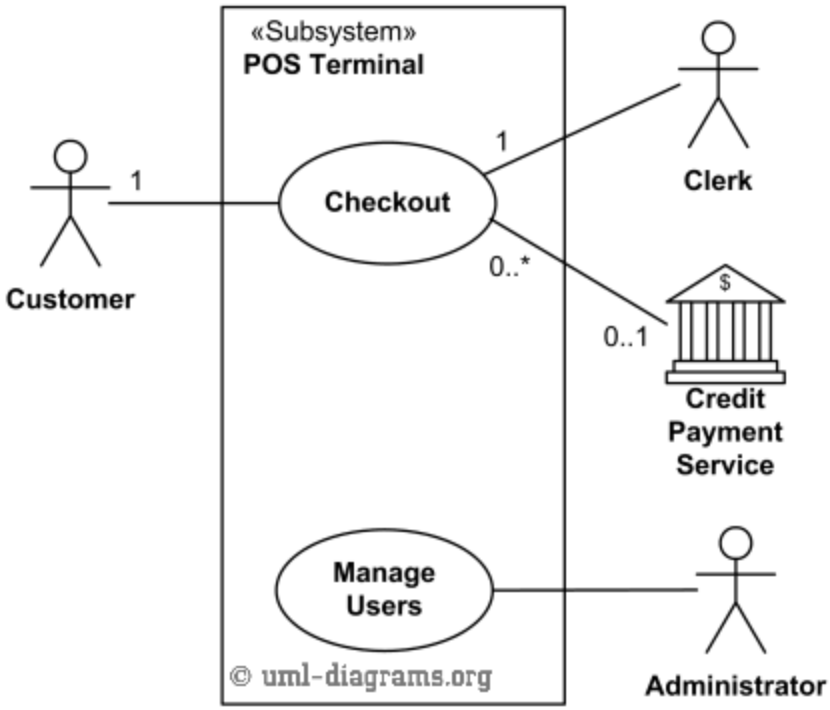
An example of use case diagram for airport check-in and security screening.

Q11. USE CASE DIAGRAM FOR RETAIL POINT OF SALE SYSTEM

UML Use Case Diagram Example

An example of UML use case diagram for **Point of Sale (POS) Terminal** or Checkout. A retail POS system typically includes a computer, monitor, keyboard, barcode scanners, weight scale, receipt printer, credit card processing system, etc. and POS terminal software.

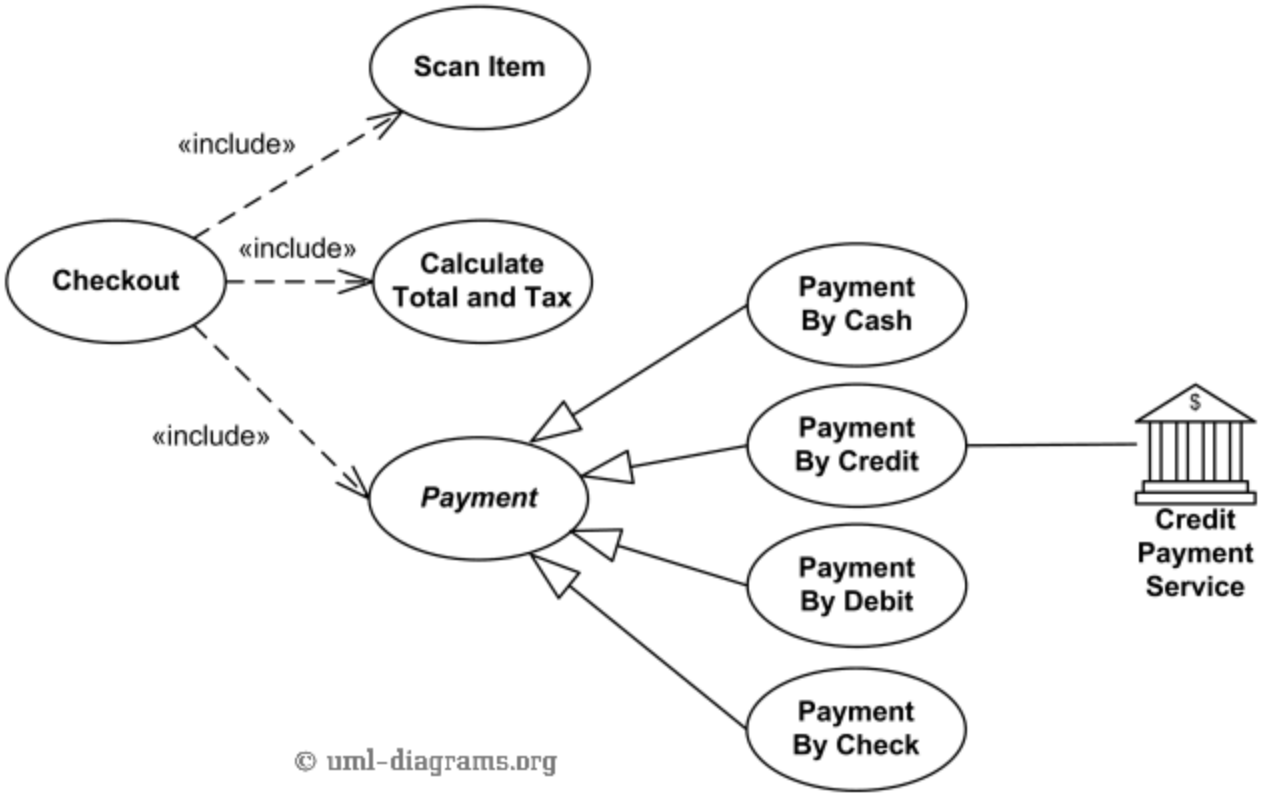
Checkout use case involves Customer, Clerk and Credit Payment Service actors and includes scanning items, calculating total and taxes, payment use cases.



Top level UML use cases for Point of Sales Terminal (POS).

Checkout use case requires Customer actor, hence the 1 multiplicity of Customer. Clerk can only participate in a single Checkout use case. Credit Payment Service can participate with many Checkout use cases at the same time. Checkout use case may not need Credit Payment Service (for example, if payment is in cash), thus the 0..1 multiplicity.

Checkout use case is an example of a large and complex use case split into several use cases each describing some logical unit of behavior. Note, that including use case becomes incomplete by itself and requires the included use cases to be complete.



Checkout use case includes Scan Item, Calculate Total and Tax, and Payment use cases.

Payment use case is represented using generalization relationship. It means that only one specific type of payment is accepted - either by cash, or by credit, debit, or with check. An alternative to such representation could be to use include relationship so that not just single but several forms of payment could be accepted from the same client during checkout.

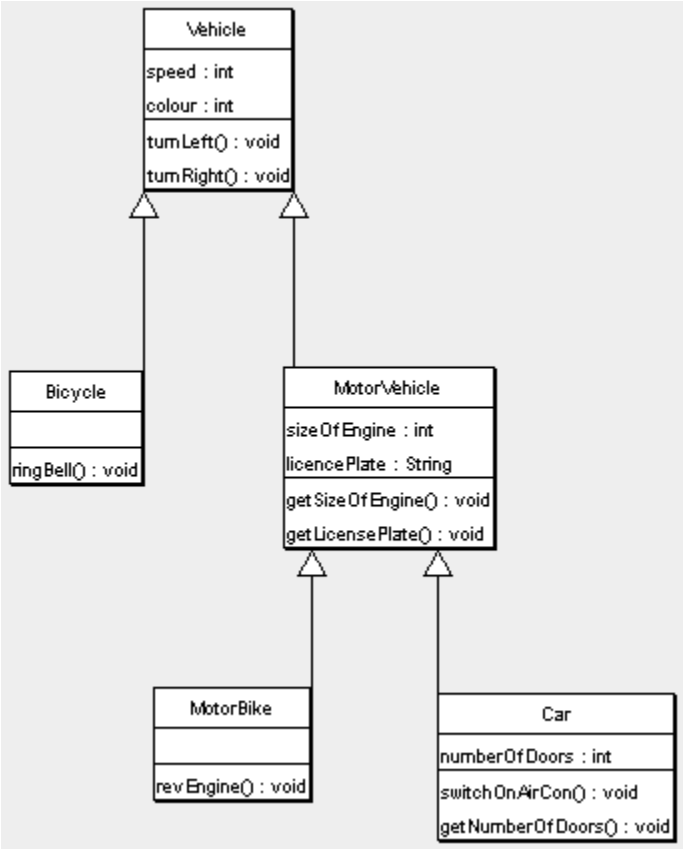
UML Class Diagrams and Examples

Here are some examples of UML class diagrams and explanations of their contents. See if you can interpret the last diagram yourself. Refer to Handout #2 for help with the UML syntax.

Q12. CLASS DIAGRAM FOR VEHICLE

Example #1: Inheritance – Vehicles

This diagram shows an *inheritance hierarchy* – a series of classes and their subclasses. Its for an imaginary application that must model different kinds of vehicles such as bicycles, motor bike and cars.



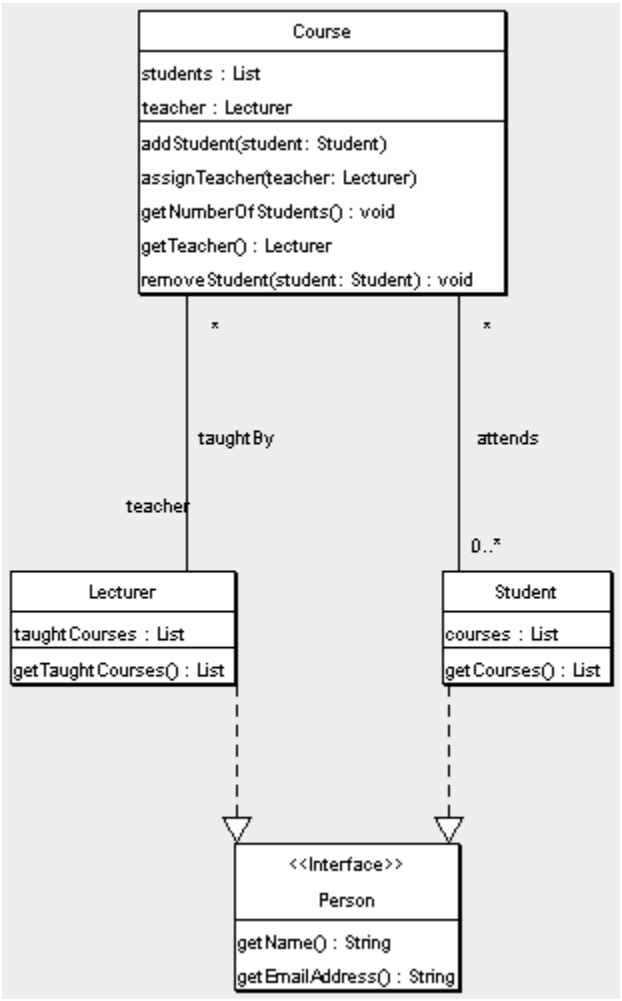
Notes

- All Vehicles have some common attributes (speed and colour) and common behaviour (turnLeft, turnRight)
- Bicycle and MotorVehicle are both kinds of Vehicle and are therefore shown to inherit from Vehicle. To put this another way, Vehicle is the superclass of both Bicycle and MotorVehicle
- In our model MotorVehicles have engines and license plates. Attributes have been added accordingly, along with some behaviour that allows us to examine those attributes
- MotorVehicles is the base class of both MotorBike and Car, therefore these classes not only inherit the speed and colour properties from Vehicle, but also the additional attributes and behaviour from MotorVehicle
- Both MotorBike and Car have additional attributes and behaviour which arespecific to those kinds of object.

Q1. CLASS DIAGRAMS STUDENT COURSES AND LECTURER

Example #2: Relationships – Students and Courses

This example demonstrates relationships between classes. It's from an imaginary application that models university courses.



Notes

- Each Course object maintains a list of the students on that course and the lecturer who has been assigned to teach that course
- The Course object has behaviour that allow the adding and removing of students from the course, assigning a teacher, getting a list of the currently assigned students, and the currently assigned teacher.

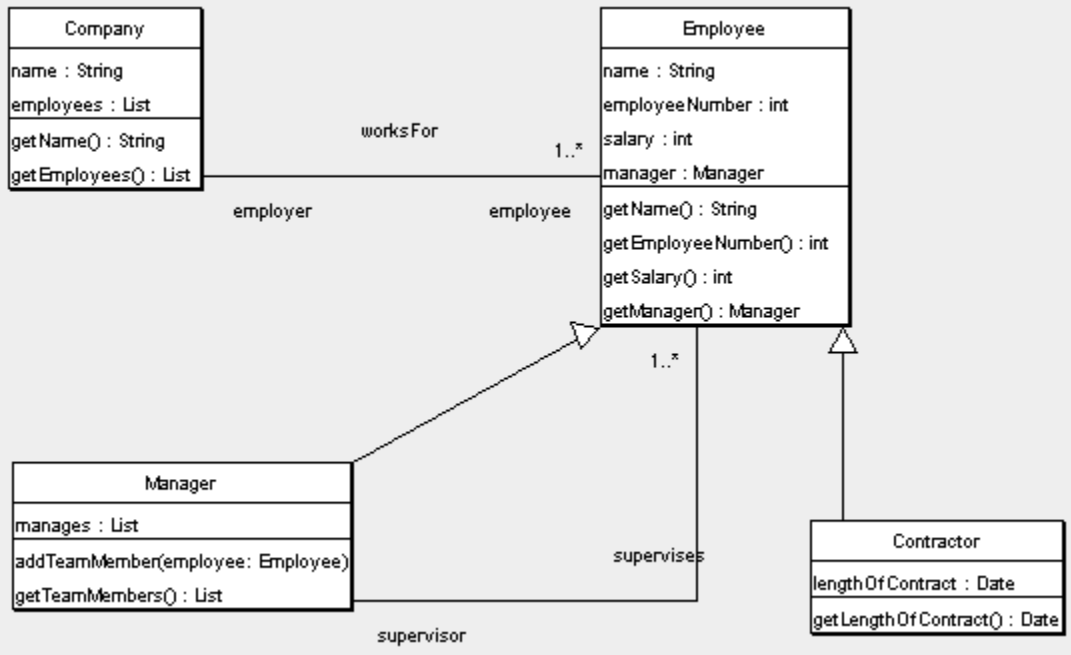
- Teachers are modelled as Lecturer objects. As a lecturer may teach more than one course there is an association between Course and Lecturer. The “taughtBy” relationship shows that a Course only has a single teacher, but that a lecturer may teach several Courses.
- Each Lecturer object also maintains a list of the Courses that it teaches.
- There is a similar relationship between Course and Student. A course is attended by zero or more Students, and a Student may attend multiple courses.

This example also demonstrates the use of interfaces. The diagram shows a Person interface that stipulates that objects conforming to this interface will have a getName and getEmailAddress methods. Both Lecturer and Student are shown to be types of Person.

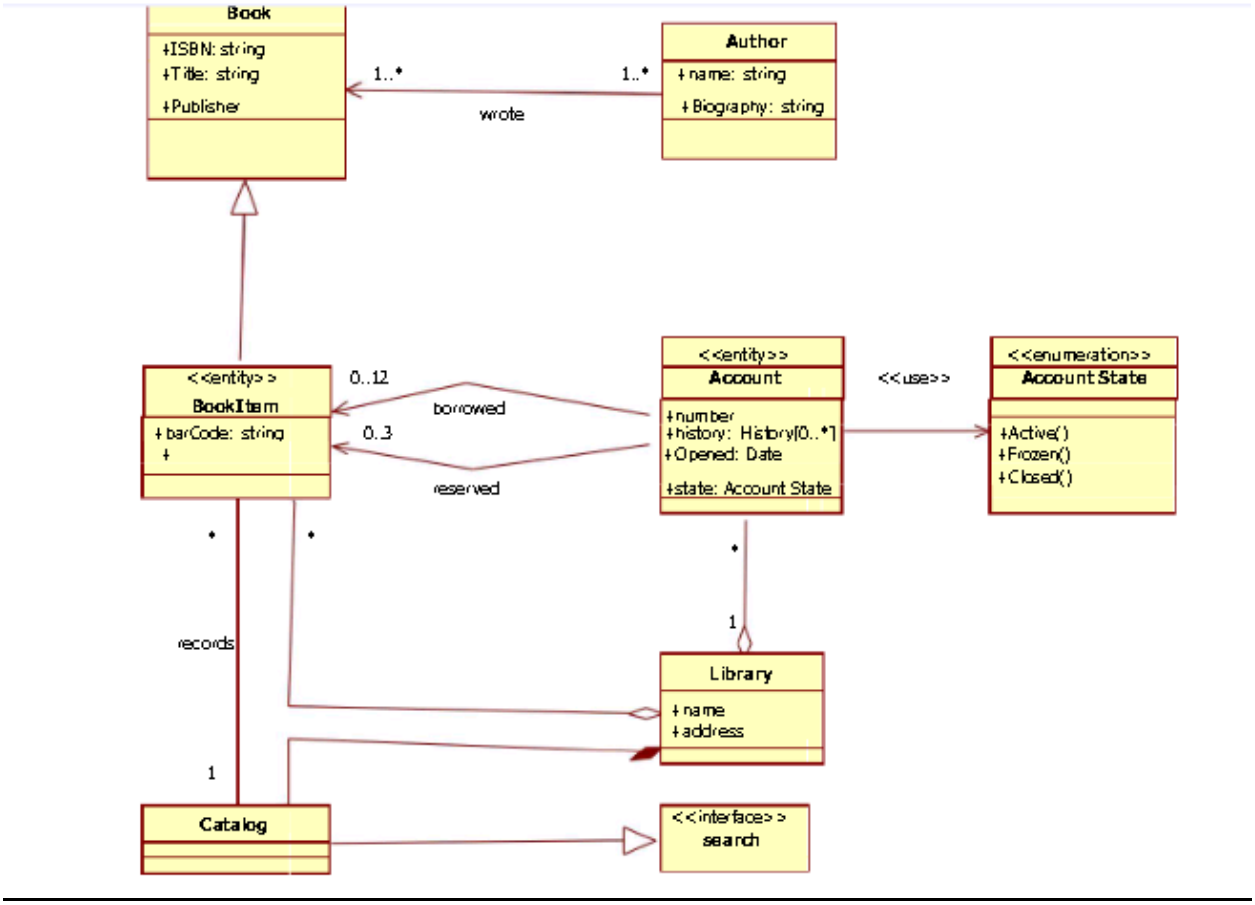
Q2. CLASS DIAGRAM FOR PAYROLL REPORTING SYSTEM

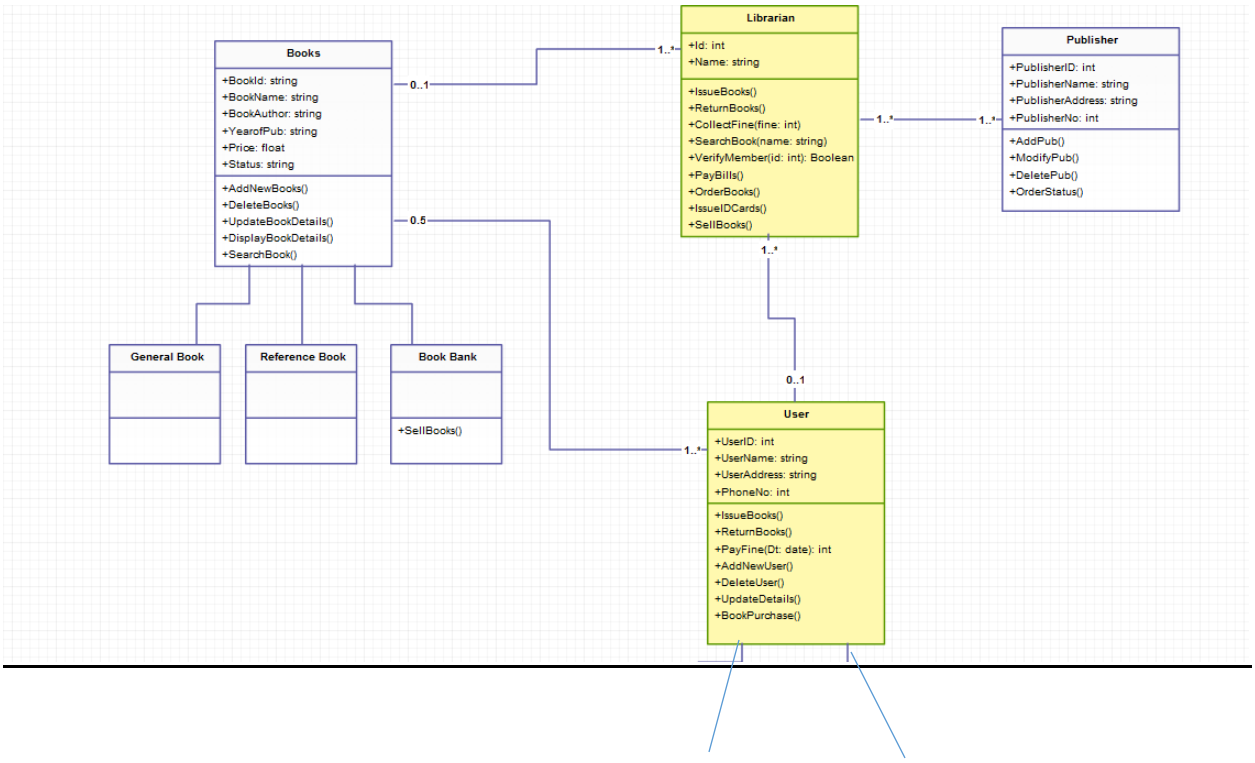
Example #3: Mixed Example – Company, Employee, Manager

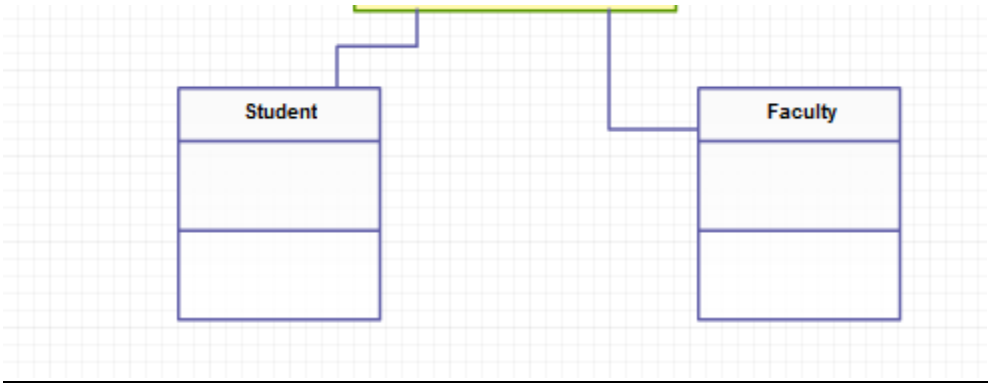
This example is from a system that models companies, e.g. for **a payroll or reporting system**. See if you can interpret it yourself!



Q3. CLASS DIAGRAM FOR LIBRARY MANAGEMENT SYSTEM


Book
+ISBN: string
+Title: string
+Publisher
1..*
1..*
wrote
Author
+name: string
+Biography: string
<<entity>>
BookItem
+barCode: string
+
0..12
borrowed
0..3
reserved
<<entity>>
Account
+number
+history: History[0..*]
+Opened: Date
+state: Account State
<<use>>
<<enumeration>>
Account State
+Active()
+Frozen()
+Closed()
records
1
Library
+name
+address
Catalog
<<interface>>
search


Books
+BookId: string
+BookName: string
+BookAuthor: string
+YearofPub: string
+Price: float
+Status: string
+AddNewBooks()
+DeleteBooks()
+UpdateBookDetails()
+DisplayBookDetails()
+SearchBook()
0..1
1..*
Librarian
+Id: int
+Name: string
+IssueBooks()
+ReturnBooks()
+CollectFine(fine: int)
+SearchBook(name: string)
+VerifyMember(id: int): Boolean
+PayBills()
+OrderBooks()
+IssueIDCards()
+SellBooks()
1..*
1..*
Publisher
+PublisherID: int
+PublisherName: string
+PublisherAddress: string
+PublisherNo: int
+AddPub()
+ModifyPub()
+DeletePub()
+OrderStatus()
0.5
1..*
1..*
0..1
General Book
Reference Book
Book Bank
+SellBooks()
User
+UserID: int
+UserName: string
+UserAddress: string
+PhoneNo: int
+IssueBooks()
+ReturnBooks()
+PayFine(Dt: date): int
+AddNewUser()
+DeleteUser()
+UpdateDetails()
+BookPurchase()



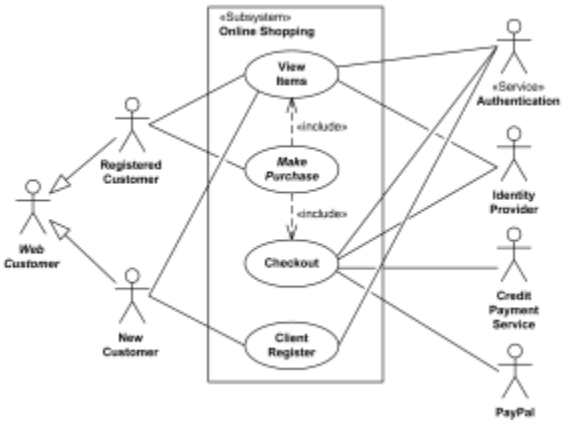
UML Diagram Examples
Online Shopping

Here we provide several UML diagrams of different types, all part of a simple fabricated **Online Shopping** model:

Online shopping UML use case diagram examples

Purpose: Provide top level use cases for a web customer making purchases online.

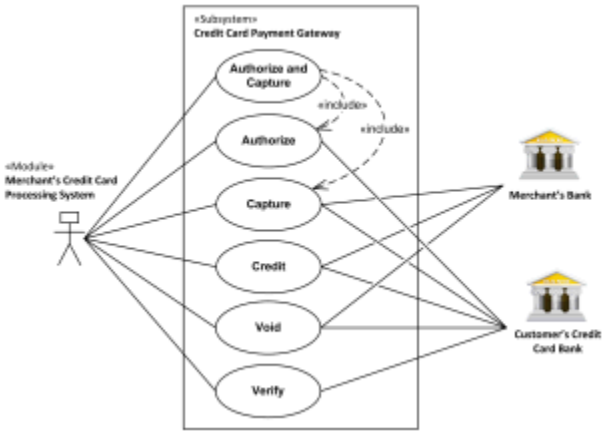
Summary: Web customer actor uses some web site to make purchases online. Top level use cases are **View Items**, **Make Purchase** and **Client Register**.



Credit card processing system

Purpose: Define major use cases for a **credit card processing system** (**credit card payment gateway**).

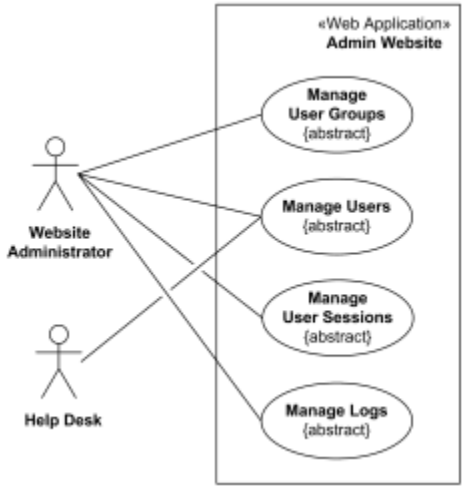
Summary: The merchant submits a credit card transaction request to the credit card payment gateway on behalf of a customer. Bank which issued customer's credit card is actor which could approve or reject the transaction. If transaction is approved, funds will be transferred to merchant's bank account.

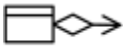


[Website administration]

Purpose: Website management or administration UML use case diagrams example.

Summary: Website Administrator actor could manage user groups, users, user sessions, and logs. Help Desk staff uses a subset of functions available to the Website Administrator.

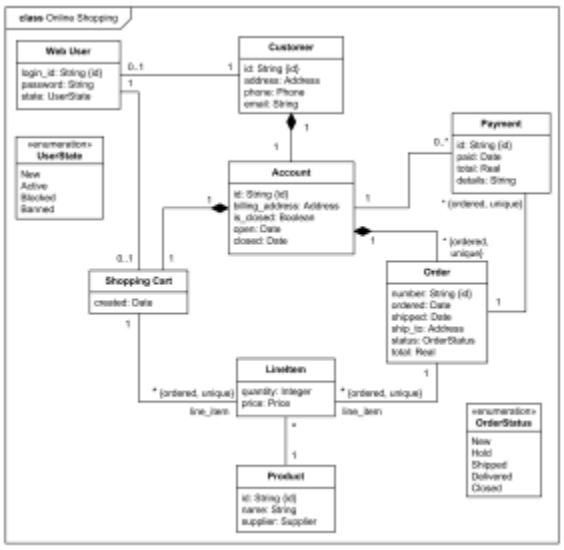


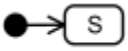


[Online shopping domain model]

Purpose: Show some domain model for online shopping - Customer, Account, Shopping Cart, Product, Order, Payment.

Summary: Example of a UML class diagram representing online shopping domain. Each customer could have some web user identity. Web user could be in one of several states and could be linked to a shopping cart.

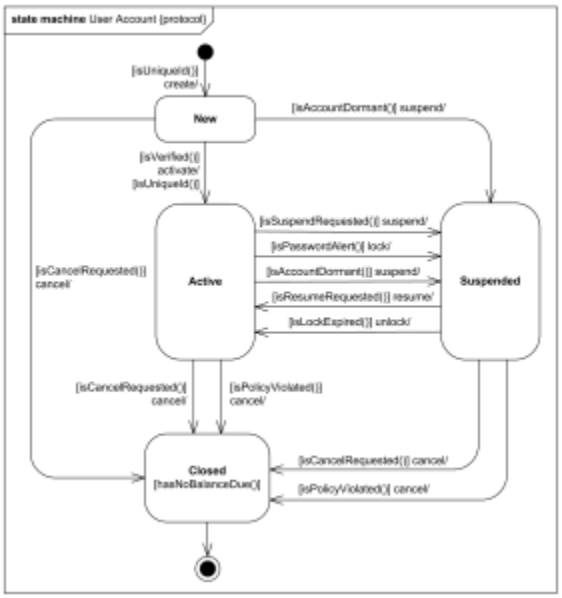




User account UML state machine diagram example

Purpose: An example of user account life cycle in the context of online shopping, and shown as UML protocol state machine diagram.

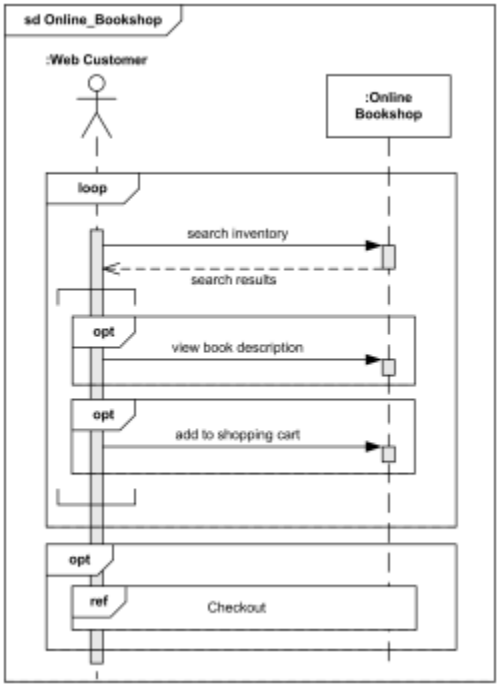
Summary: Every company having customers maintains customer accounts and supports a complete life cycle of the account from its creation until it is closed. There are differences in what are the stages (states) in the account's life cycle, and what are conditions or events causing account to change its state.



Online bookshop UML sequence diagram

Purpose: An example of high level UML sequence diagram for online bookshop.

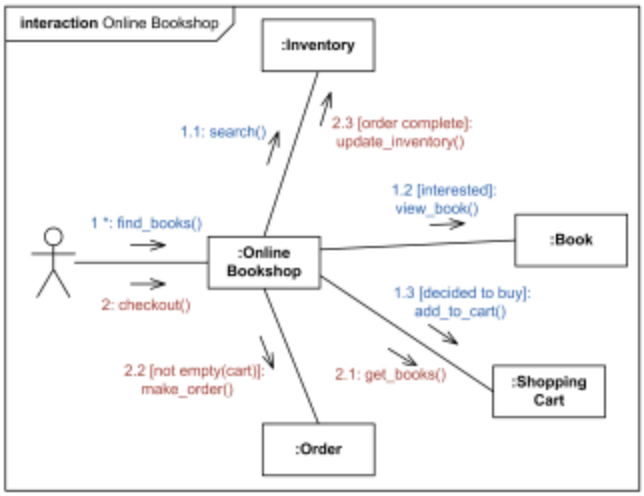
Summary: Online customer can search book catalog, view description of a selected book, add book to shopping cart, do checkout.

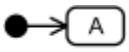


Online bookshop communication diagram

Purpose: An example of UML communication diagram for Online Bookshop.

Summary: Web customer actor can search inventory, view and buy books online.

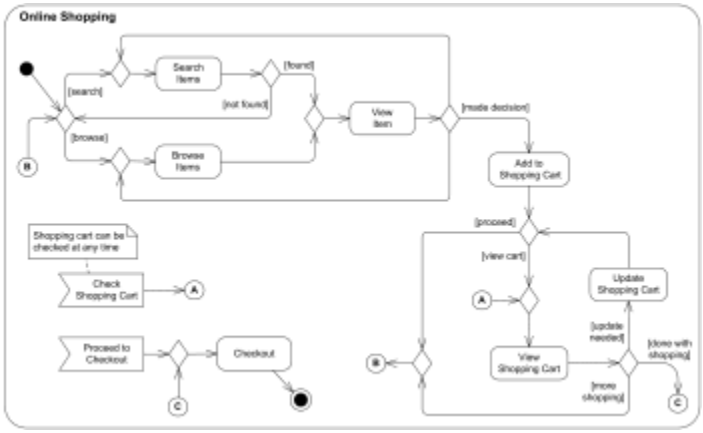


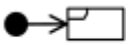


Online shopping activity diagram

Purpose: An example of activity diagram for online shopping.

Summary: Online customer can browse or search items, view specific item, add it to shopping cart, view and update shopping cart, checkout. User can view shopping cart at any time. Checkout includes user registration and login.

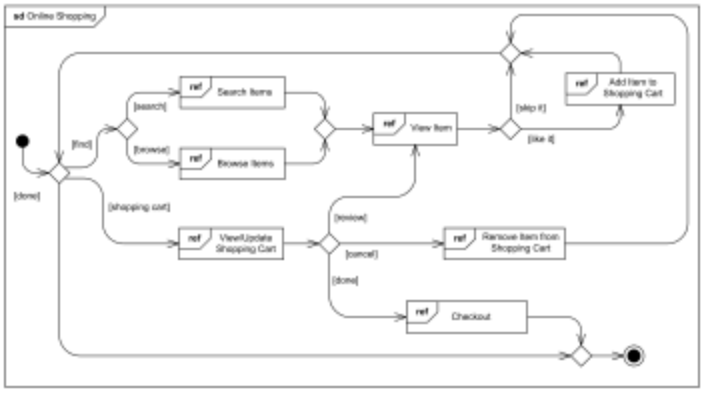


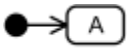


Online shopping interaction overview diagram

Purpose: An example of interaction overview diagram for online shopping.

Summary: Customer may search or browse items, add or remove items from shopping cart, do checkout.

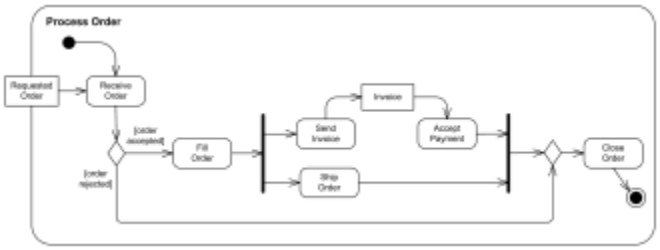


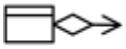


Business flow - process purchase order

Purpose: An example of business flow UML activity diagram to process purchase order.

Summary: Requested order is input parameter of the activity. After order is accepted and all required information is filled in, payment is accepted and order is shipped.

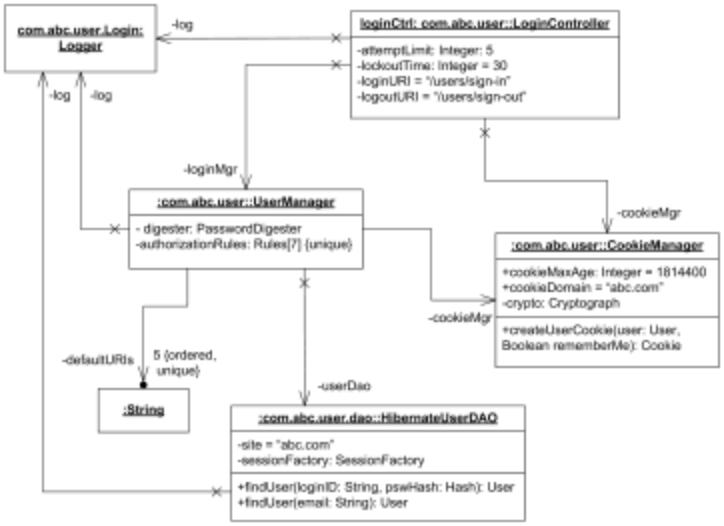


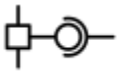


Web application Login Controller object diagram

Purpose: An example of UML object diagram which shows some runtime objects involved into login process for a web user.

Summary: An instance of Login Controller class is associated with instances of User Manager, Cookie Manager, and Logger. Login Controller, User Manager, and Hibernate User DAO (Data Access Object) share a single instance of Logger.

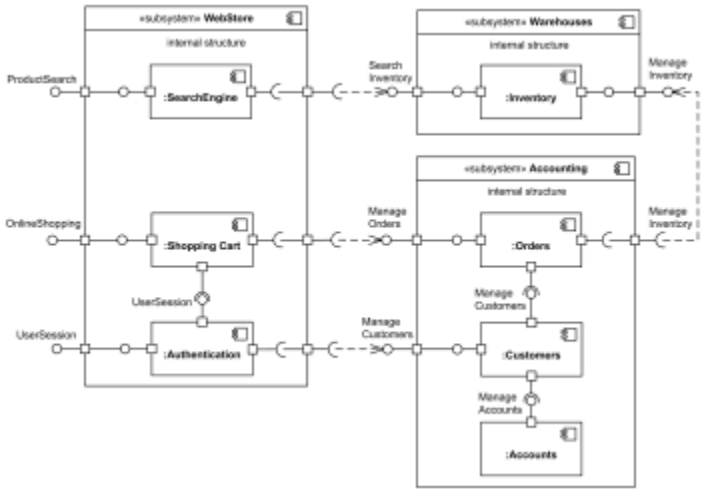


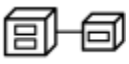


Online shopping component diagram

Purpose: An example of a component diagram for online shopping.

Summary: The diagram shows "white-box" view of the internal structure of three related subsystems - WebStore, Warehouses, and Accounting. **WebStore** subsystem contains three components related to online shopping - **Search Engine**, **Shopping Cart**, and **Authentication**. **Accounting** subsystem provides two interfaces - **Manage Orders** and **Manage Customers**. **Warehouses** subsystem provides two interfaces **Search Inventory** and **Manage Inventory** used by other subsystems.

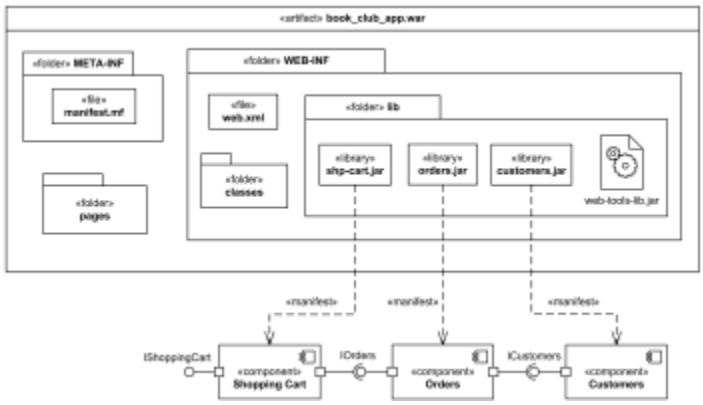




Online shopping web application manifestation

Purpose: An example of manifestation diagram for a web application.

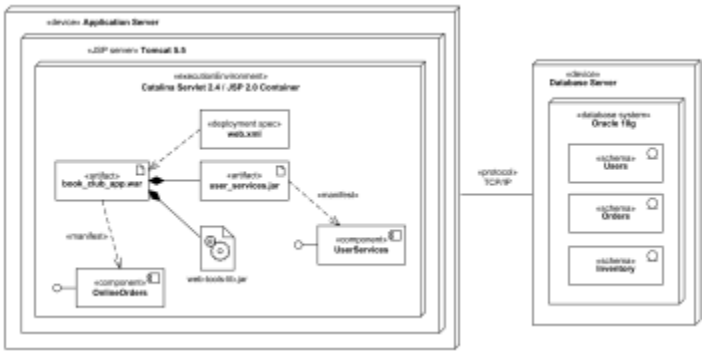
Summary: Web application archive artifact book_club_app.war contains several files, folders and subfolders. Several components are shown on the diagram manifested (implemented) by jar archive files.



Online shopping web application UML deployment diagram

Purpose: An example of UML deployment diagram for a J2EE web application.

Summary: Book club web application artifact book_club_app.war is deployed on Catalina Servlet 2.4 / JSP 2.0 Container which is part of Apache Tomcat 5.5 web server.

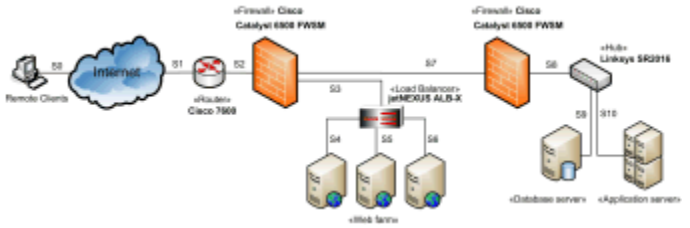


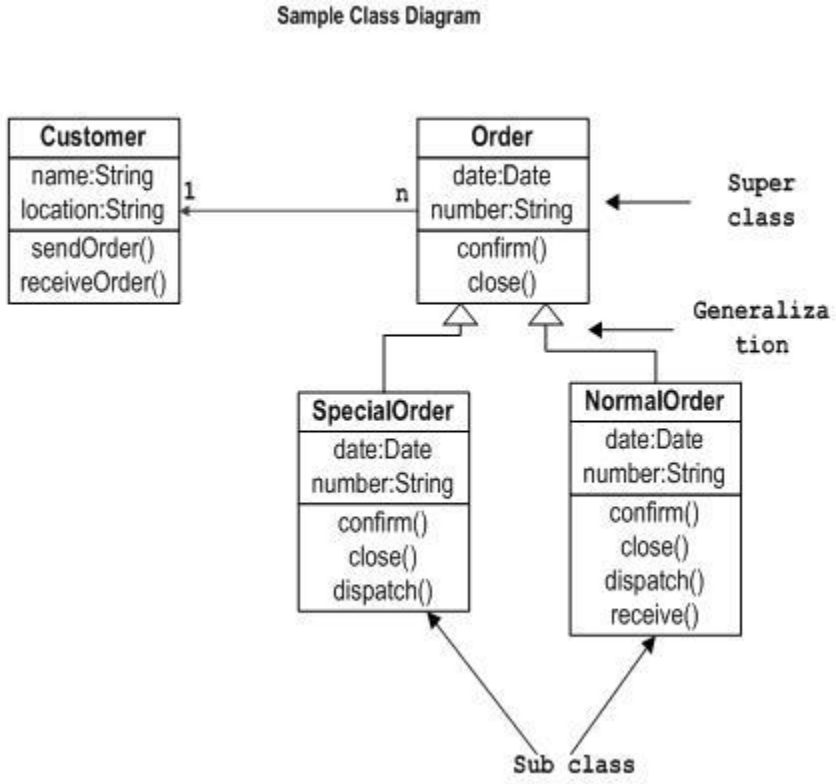
Web application network diagram example

Purpose: UML provides no special kind of diagram to describe logical or physical **network architecture** of the designed or existing system. Deployment diagrams could be used for this purpose with elements limited mostly to devices with neither artifacts nor actual deployments shown.

Summary: This example of the network diagram shows network architecture with configuration usually called "two firewall demilitarized zone (DMZ)". DMZ is a host or network segment located in a "neutral zone" between the Internet and an organization’s intranet (private network).

It prevents outside users from gaining direct access to an organization's internal network while not exposing a web, email or DNS server directly to the Internet.

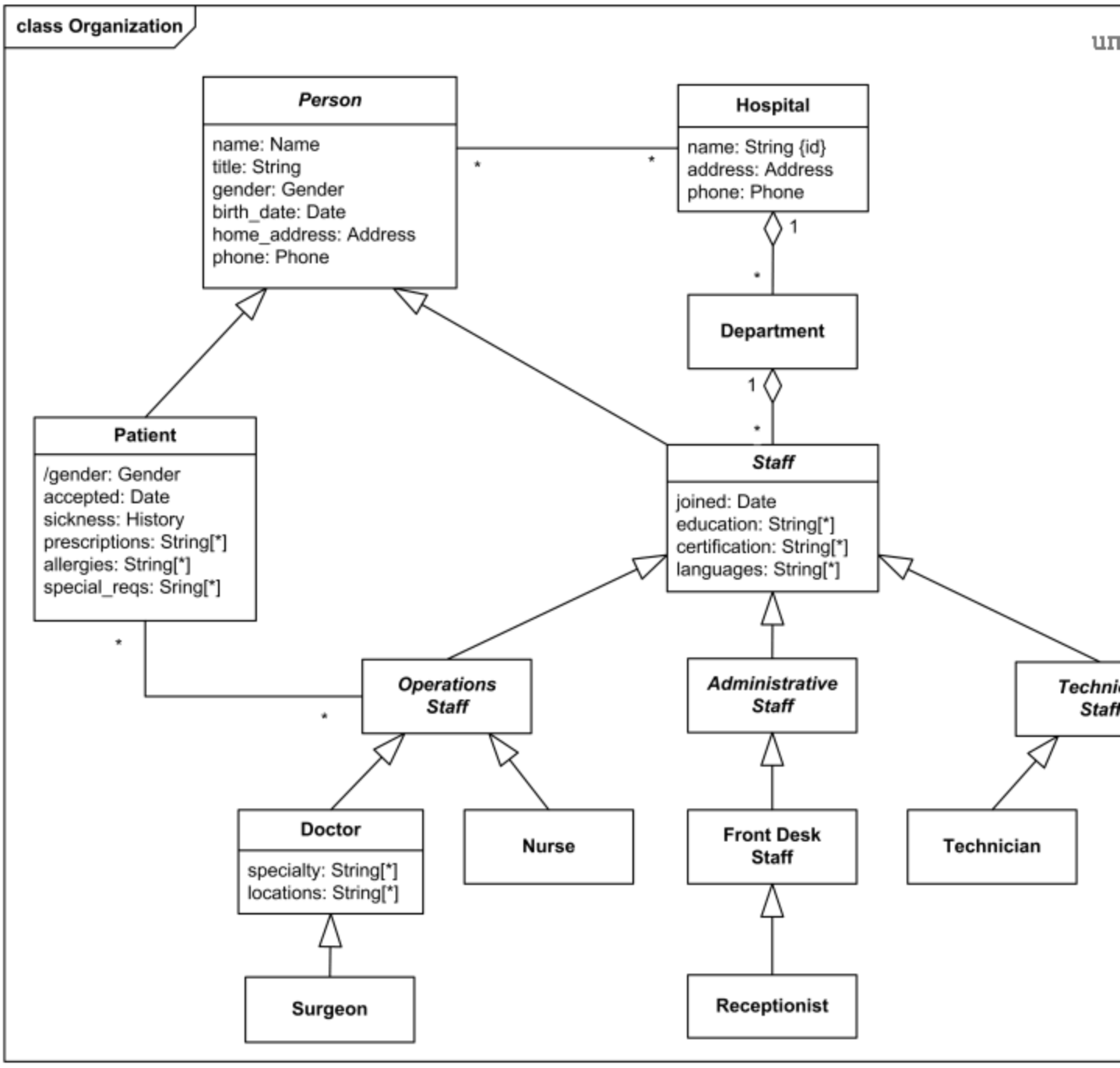




This is an example that models "**Hospital Management**". The **ward** object of this system has attributes such as name, patient-gender and capacity. Note that patient-gender is a gender type which is an enumeration containing enums male and female. The system also has **Patient** entity with attributes such as patient_id,

admitted, sickness _history, prescriptions, special_reqs and allergies and gender which is again a gender enumeration type. And operations such as getPatient() and deletePatient(Patient_id).Ward is a division of a **hospital** object having attributes such as name address and phone number. In hospital there are number of wards each of which may be empty or have one or more patient. Each ward has unique name. This ward is shared by patients who need a similar kind of care. Each patient is on a single ward. The system also has **Doctor** entity which is further classified into **Consultant Doctor** and **Junior Doctor**. The doctors in the hospital are organized into **Teams** entity with attribute team_name. Each team can have two or more doctors. Each patient is under the care of a single team of doctors. A patient may be treated by any number of doctors but all the doctors must belong to same team that cares for the patient. Note that team is own by the hospital.

This is an example of a hospital domain model diagram. The domain model for the **Hospital Management System** is represented by several **class diagrams**.
Purpose: *Domain model for a hospital to show and explain hospital structure, staff, relationships with patients, and patient treatment terminology.*

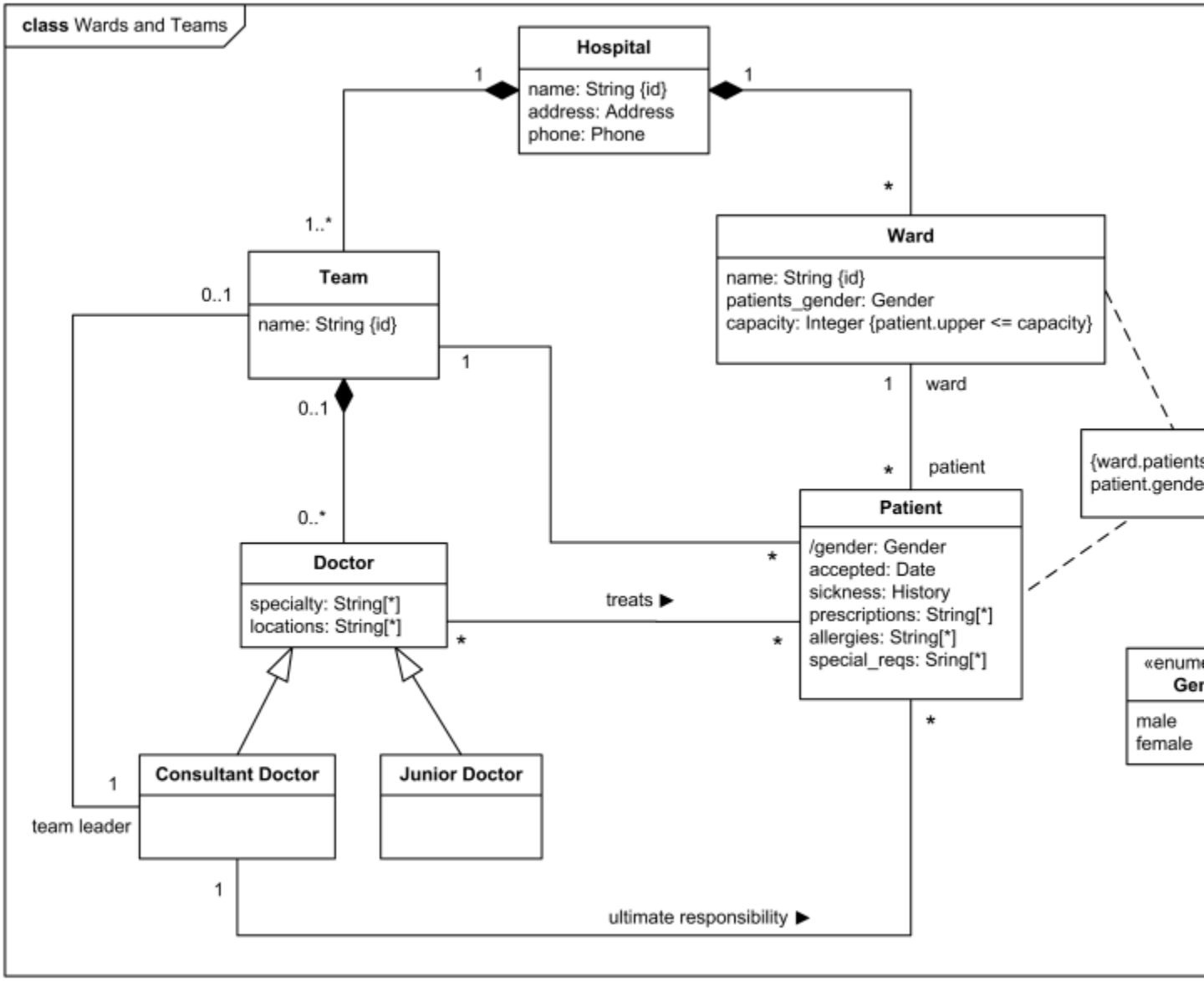


Hospital Organization Domain model - Patient, Hospital, Staff - Operations, Administrative, Technical.

Ward is a division of a hospital or a suite of rooms shared by patients who need a similar kind of care. In a hospital, there are a number of wards, each of which may be empty or have on it one or more **patients**. Each ward has a unique name. Diagram below shows it using **{id}** modifier for ward's name.

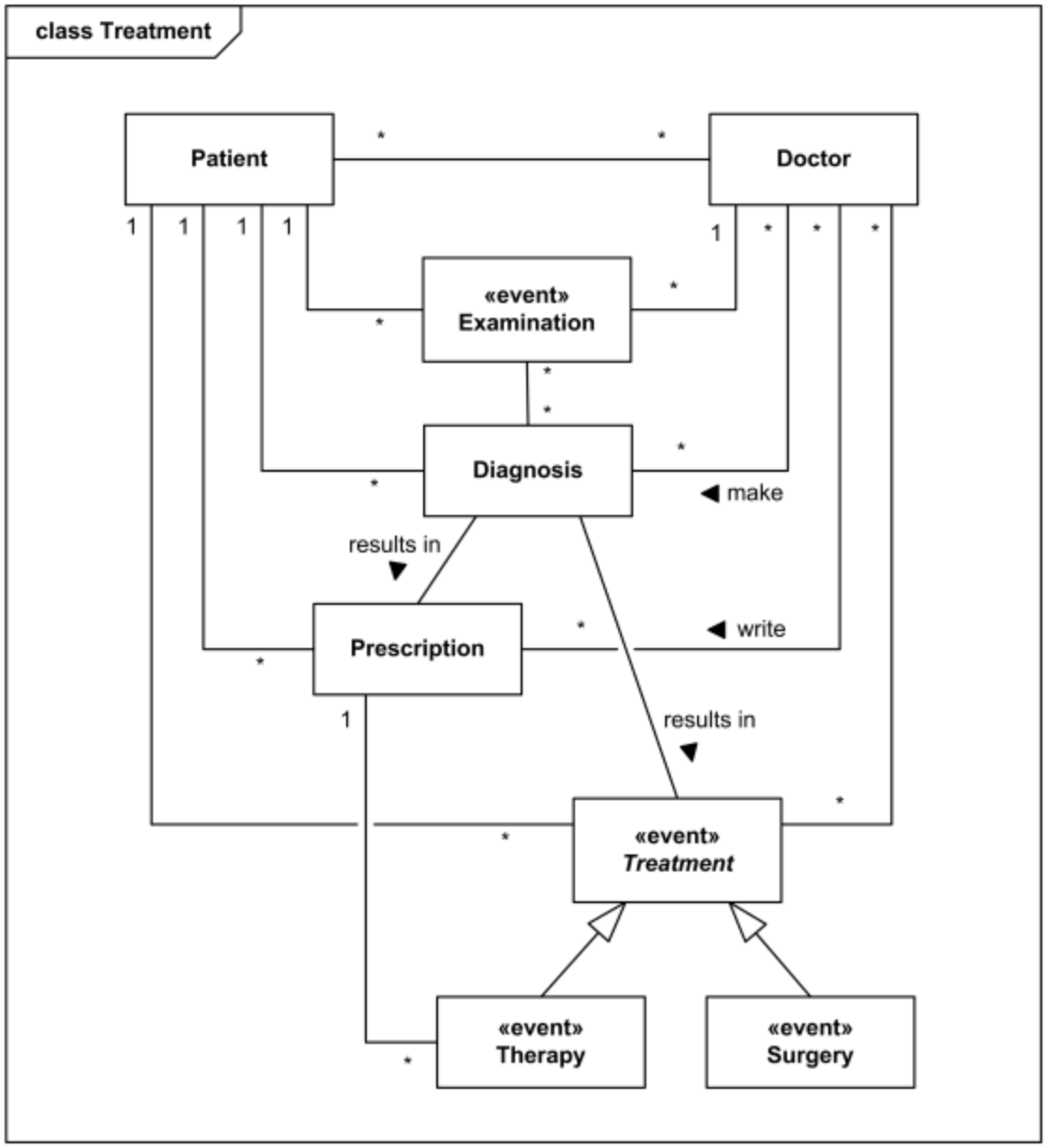
Wards are differentiated by **gender** of its patients, i.e. male wards and female wards. A ward can only have patients of the gender admitted to it. Gender is shown as enumeration. Ward and patient have constraint on Gender.

Every ward has a fixed capacity, which is the maximum number of patients that can be on it at one time (i.e. the capacity is the number of beds in the ward). Different wards may have different capacities.
The doctors in the hospital are organised into **teams** (also called **firms**). Each team has a unique name or code (e.g. Orthopaedics or Pediatrics) and is headed by a **consultant doctor** (in the UK, Republic of Ireland, and parts of the Commonwealth) or **attending physician** (also known as staff physician) (in the United States). Consultant doctor or attending physician is the senior doctor who has completed all of his or her specialist training, residency and practices medicine in a clinic or hospital, in the specialty learned during residency. She or he can supervise fellows, residents, and medical students. The rest of the team are all junior doctors. Each doctor could be a member of no more than one team.

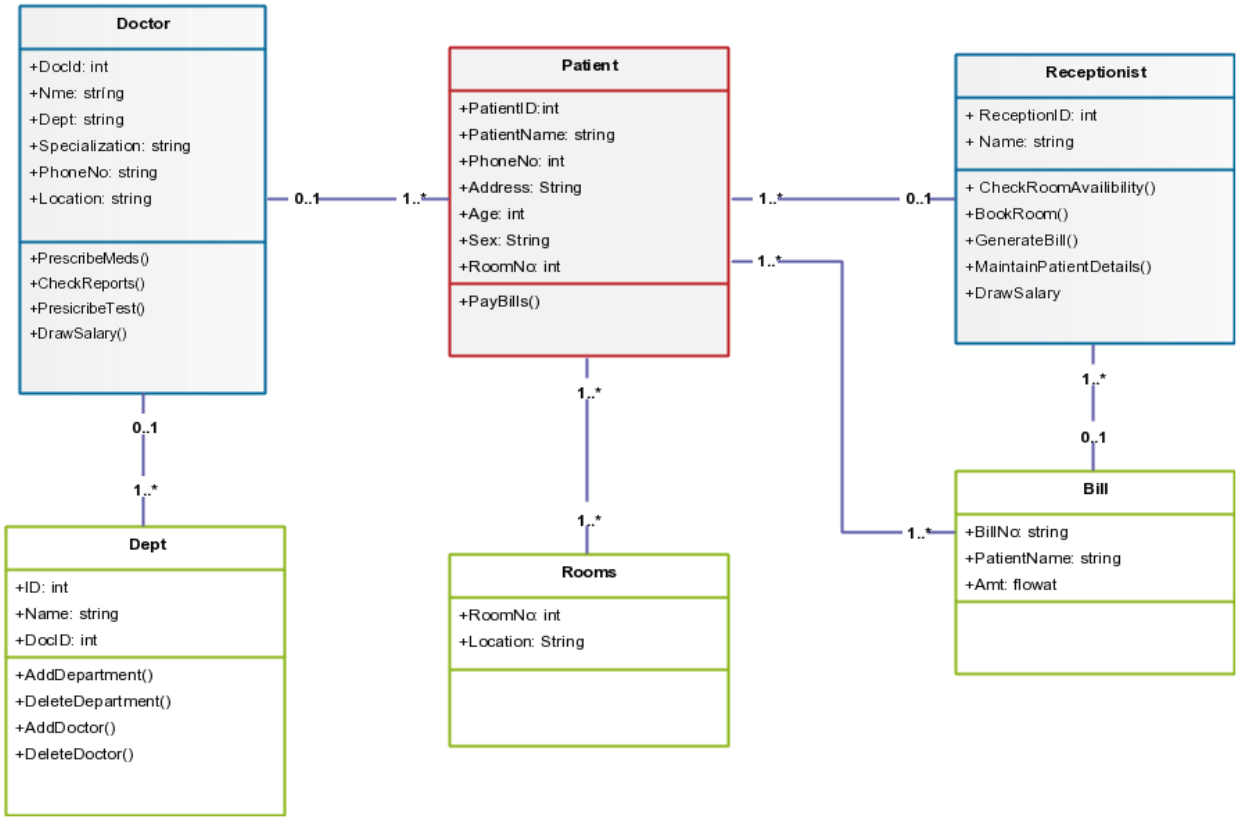


Hospital wards, teams of doctors, and patients.
Each **patient** is on a single ward and is under the care of a single team of doctors. A patient may be treated by any number of doctors but they must all be in the team that cares for the patient. A

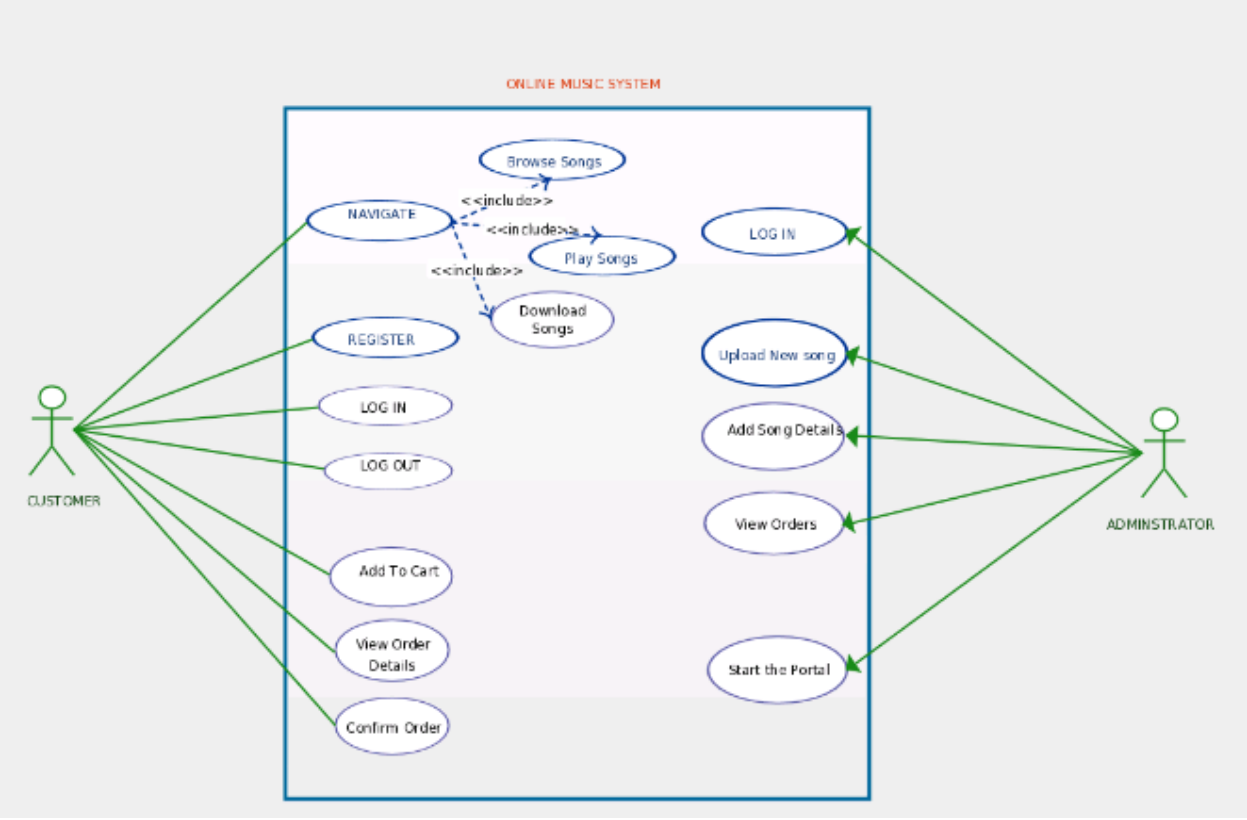
doctor can treat any number of patients. The team leader accepts ultimate responsibility, legally and otherwise, for the care of all the patients referred to him/her, even with many of the minute-to-minute decisions being made by subordinates.



Domain model - Patient, Doctors and Treatments



ONLINE MUSIC PORTAL USE CASE DIAGRAM



Refer

http://www.uml-diagrams.org

http://creately.com/blog/examples/class-diagram-templates/