I T P 4 9 0 9 - O O T - M A I N - 2 0 1 9 - M a y

Section A (40 marks)
This section contains THREE questions. Answer ALL questions.

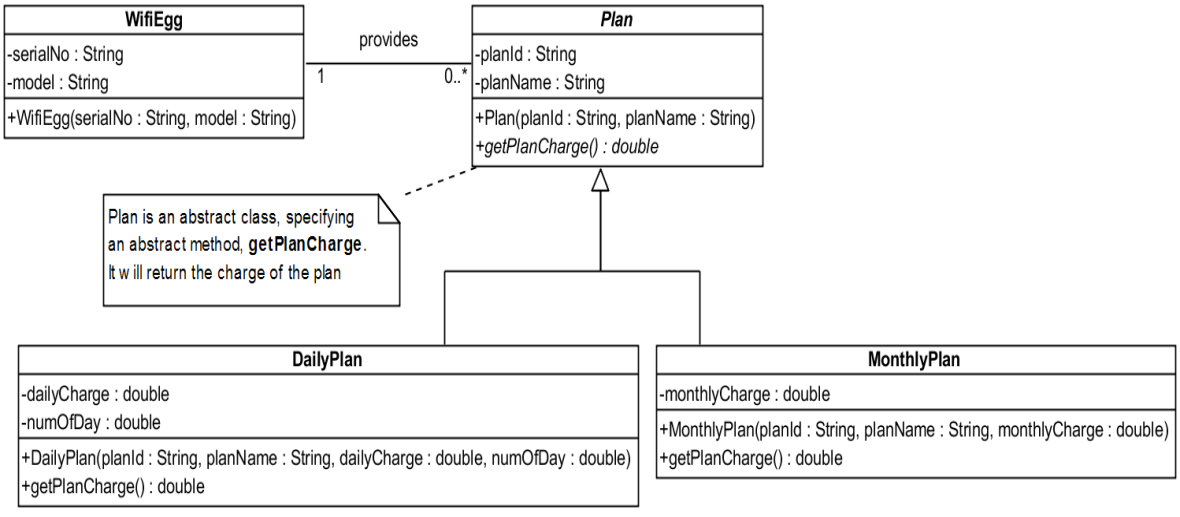
A1. Consider the following scenario:

A café has staff with different responsibilities. Each staff has a unique staff identification, name and address. A staff can be a manager, cashier and chef. The manager will physically count the items in the warehouse. Each item has a unique identification, name and expiry date. Each item will have more than one supplier. The chef will review a list of existing items and suggest any new items to the manager. The manager will order the items which level is below a level. The cashier is responsible for scanning the barcode on an item and generates receipts for customers. Each receipt has a receipt number, receipt date and total amount.

Draw a class diagram to model the classes and the relationships between classes for the above problem statement. You are required to show the name and multiplicity of each association in your diagram. You need to show all the required attributes in each class based on the given information. You should also structure the classes with inheritance if possible. Note: You **do not need** to add qualifications and/or association classes for this question.

[12 marks]

A2. The following class diagram shows the relationship between **WifiEgg** class and the related **plan** class.



QUESTION A2 CONTINUES ON THE NEXT PAGE

QUESTION A2 CONTINUES FROM THE PREVIOUS PAGE

```
public class Test {
    public static void main(String[] args) {

        WifiEgg egg1 = new WifiEgg("S123", "4.5G");
        WifiEgg egg2 = new WifiEgg("S124", "4G");

        Plan[] plans = new Plan[3];
        plans[0] = new MonthlyPlan("S001", "WorldWide ", 286);

        plans[1] = new DailyPlan("D001", "WorldWide", 24, 7);
        plans[2] = new MonthlyPlan("S001", "Local", 286);

        // The addPlan method is a method in the WifiEgg class
        // It is used to add a plan to a WifiEgg
        egg1.addPlan(plans[0]);
        egg1.addPlan(plans[1]);
        egg2.addPlan(plans[2]);

        // The setWifiEgg method is a method in the Plan class
        // It is used to set the wifiEgg for a Plan
        plans[0].setWifiEgg(egg1);
        plans[1].setWifiEgg(egg1);
        plans[2].setWifiEgg(egg2);

        double totalCharges = 0;
        Enumeration e = egg1.getPlans();
        // The getPlans method is a method in the WifiEgg class
        // It is used to return a set of plans from a WifiEgg
        while (e.hasMoreElements()) {
            Plan p = (Plan) e.nextElement();
            totalCharges = totalCharges + p.getPlanCharge();
            System.out.println("Plan ID: " + p.getPlanId()
                    + " plan charge: " + p.getPlanCharge());
        }
        System.out.println();
        System.out.println("Total charge for WifiEgg (" + egg1.getSerialNo()
           + ") is " + totalCharges);
    }
}
```

The computation method for charge in the dailyPlan class is using the following formula:

totalCharge = dailyCharge * numOfDay;

(a) Assume that *MonthlyPlan* will be completed by your teammate. Based on the given class diagram and test program *Test*.java, implement the class ***WifiEgg***, ***Plan*** and ***DailyPlan*** in Java. Your implementation should reflect the requirements in the class diagram and be able to run the Test.java program without error. [14 marks]

(b) Assume that you have successfully implemented all the classes in the given class diagram, give the expected output when the test program Test.java runs. [2 marks]

A3. Consider the following steps of a buyer to order candies through a web-based system of a candy shop.

The buyer was assumed to be registered with the system/shop already. The system displays a login screen and asks the buyer to login the system. If the login is successful, the system displays all types of candies in the shop. After the buyer has selected a type of candy, the system will then display all the available prepacked candies of the same type to the buyer. The buyer then selects a candy to buy. The system then displays the candy and payment information that the buyer selected along with two buttons "Confirm" and "Cancel". If the "Confirm" button is pressed, the system emails the online receipt and terminates. If the "Cancel" button is pressed, the system cancels the process and returns back to the login screen.

(a) Draw an activity diagram to model the procedure for a buyer to buy candies in the web. You should include the initial node, final node, the name of each action, decision node (if any) in your activity diagram. [6 marks]

(b) Draw a state machine diagram to model the procedure for a buyer to buy candies in the web. You should include the initial state, final state, the name of each ***state***, the ***transition***, guard ***condition***(if any), and ***effects*** in your state machine diagram. [6 marks]

Section B (60 marks)

This section contains THREE questions. Answer ALL questions.

The problem statement for the questions in Section B.

Ruby Coffee (RC) is a specialty coffee company aiming to give people a convenient coffee experience. RC wants to develop an online system to promote and sell their coffee subscription plan. The subscription plan allows registered customers to receive delivery of their favourite coffee based upon their chosen frequency of delivery (weekly, bi-weekly or monthly) .

A public user or registered customer can browse all coffee products in the website of RC Coffee Subscription System (RCCSS) and no login is required. They can search for different coffees by entering either the category or keywords. They can also browse all types of deliveries (weekly, bi-weekly or monthly) available to particular coffee. However, if a public user wants to subscribe coffee delivery, he/she must register an account in RCCSS by providing his/her details such as the name, email address, and delivery address. Upon successful registration of membership, a confirmation email with membership number and password will be sent to the registered customer.

To subscribe a coffee delivery, a customer first login RCCSS by entering the membership number and password. He/she then enters keywords to search for the coffee. RCCSS will return a list of coffees that are matched with the searching criteria. The customer can select one of these coffees to check its details. RCCSS will first display the packing sizes of the selected coffee. The available packing sizes are 12oz, 3lb, and 5lb. After the customer has selected the packing size, RCCSS will display a list of delivery frequencies for the customer to select. The available delivery frequencies are weekly, bi-weekly, and monthly. The customer selects the delivery frequency and then RCCSS will show the total price of the subscription. The customer can confirm the total price by clicking the “Pay” button. RCCSS will ask the customer to pay the subscription by entering the credit card number, card type, and expiry date. The customer enters the credit card details and then RCCSS forwards the payment information to an external payment gateway for approval. Upon successful approval of the credit card payment, the payment gateway will send back an approval code to RCCSS that will show a payment number and a subscription number to the customer.

If the delivery package of coffee received by a customer is damaged or wrongly packed, the customer can request for a re-delivery of that package. The customer needs to first login the system and then enters the subscription number to display his/her subscription plan. Then the customer can click the “Delivery Problem” button in the subscription plan to ask for follow-up. RCCSS will arrange a follow-up procedure that is out of the scope of this system.

Answer questions B1 to B3 based on the given problem statement.

B1. Draw a use case diagram for the RC Coffee Subscription System. Show all required use cases, actors, communication links between actors and use cases, and <include>> / <<extend>> relationships between use cases. [13 marks]

B2 (a) Perform a textual analysis on the problem statement to identify candidate classes of the RC Coffee Subscription System. List the candidate classes with the corresponding reasons for the choices of the candidate classes. [10 marks]

(b) Draw an initial class diagram to show the classes of the RC Coffee Subscription System. Show the relationships among the classes in your answer. Give appropriate names to the associations and show the multiplicities on associations.

Note: You are NOT required to show the attributes/operations of the classes and also association classes in your diagram. [9 marks]

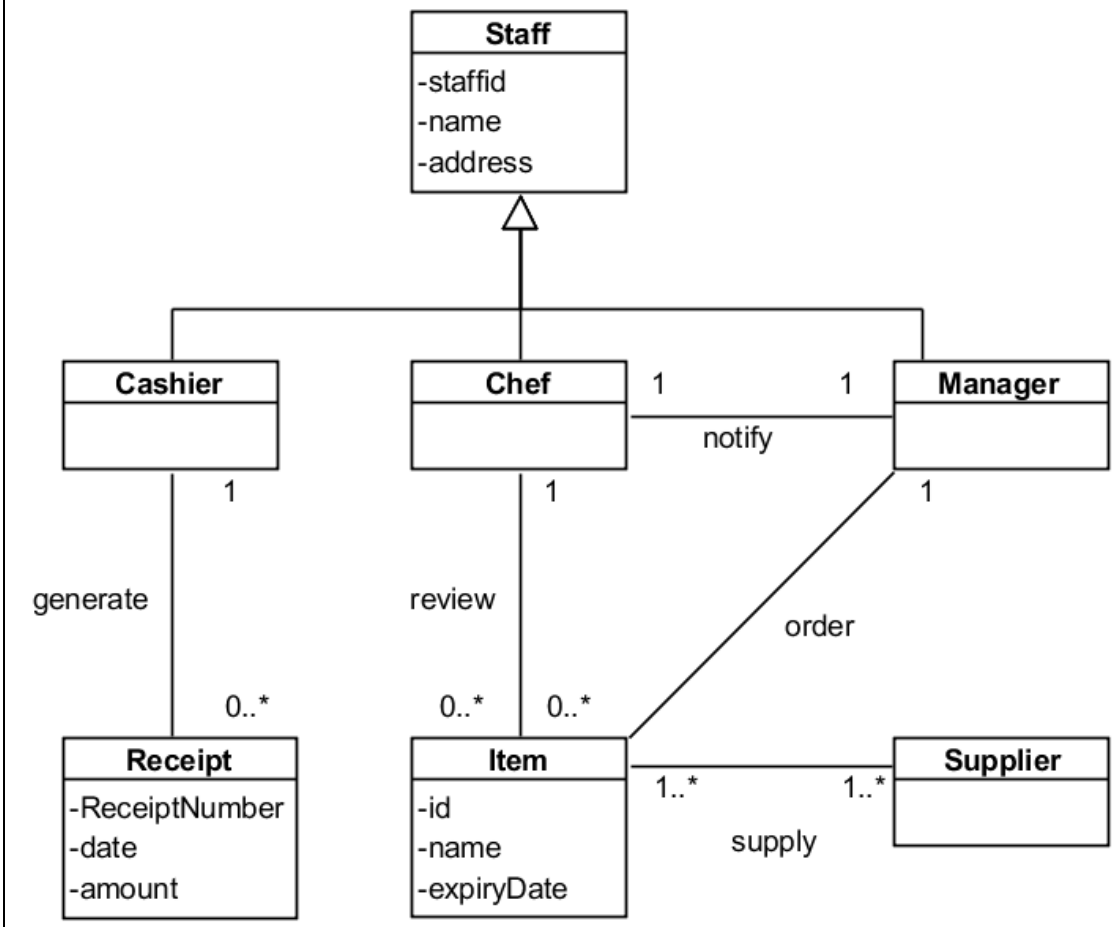
B3 Draw a three-tier sequence diagram of the Subscribe Coffee Delivery use case in the normal scenario (Assuming the customer already logged in the system).

Note: For simplicity, only ONE boundary object and ONE controller object are required for your design. [28 marks]

******* END OF PAPER *******

Suggested Solution

A1



A2

```
public class WifiEgg {
    private String serialNo;    private String model;    private Vector plans;
    public WifiEgg(String serialNo, String model) {
        this.serialNo = serialNo; this.model = model; plans = new Vector();

    }
    public String getSerialNo() {        return serialNo;    }
    public String getModel() {        return model;    }
    public void addPlan(Plan p) {        this.plans.add(p);    }
    public Enumeration getPlans() {        return plans.elements();    }
}
```

Note: one-to-many can be programmed using ArrayList or Vector or any other appropriate Java structure.

```
public abstract class Plan {
    private String planId; private String planName; private WifiEgg wifiEgg ;
    public Plan(String planId, String planName) {
        this.planId = planId;        this.planName = planName;
    }
    public String getPlanId() {    return planId;    }
    public String getPlanName() {    return planName;    }
    public void setWifiEgg(WifiEgg wifiEgg) {    this.wifiEgg = wifiEgg;    }
    public abstract double getPlanCharge();
}
```

```
public class DailyPlan extends Plan {
    private double dailyCharge;    private double numOfDay;
    public DailyPlan(String planId, String planName, double dailyCharge,
                                                    double numOfDay) {
        super(planId, planName);
        this.dailyCharge = dailyCharge;        this.numOfDay = numOfDay;
    }
    public double getDailyCharge() {    return dailyCharge;    }
    public double getNumOfDay() {    return numOfDay;    }

    @Override
    public double getPlanCharge() {    return dailyCharge * numOfDay;    }
}
```

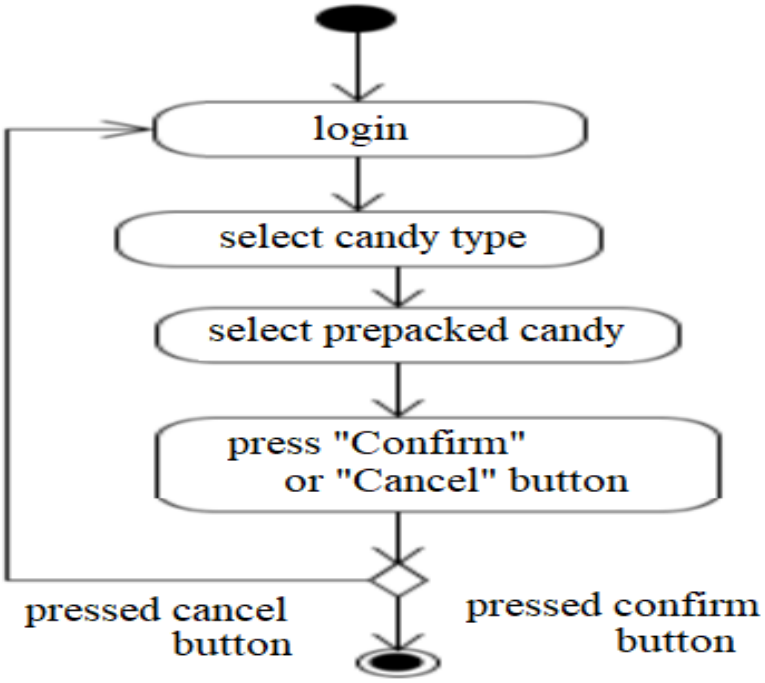
(b)

Plan ID: S001 plan charge: 286.0
Plan ID: D001 plan charge: 168.0

Total charge for WifiEgg (S123) is 454.0

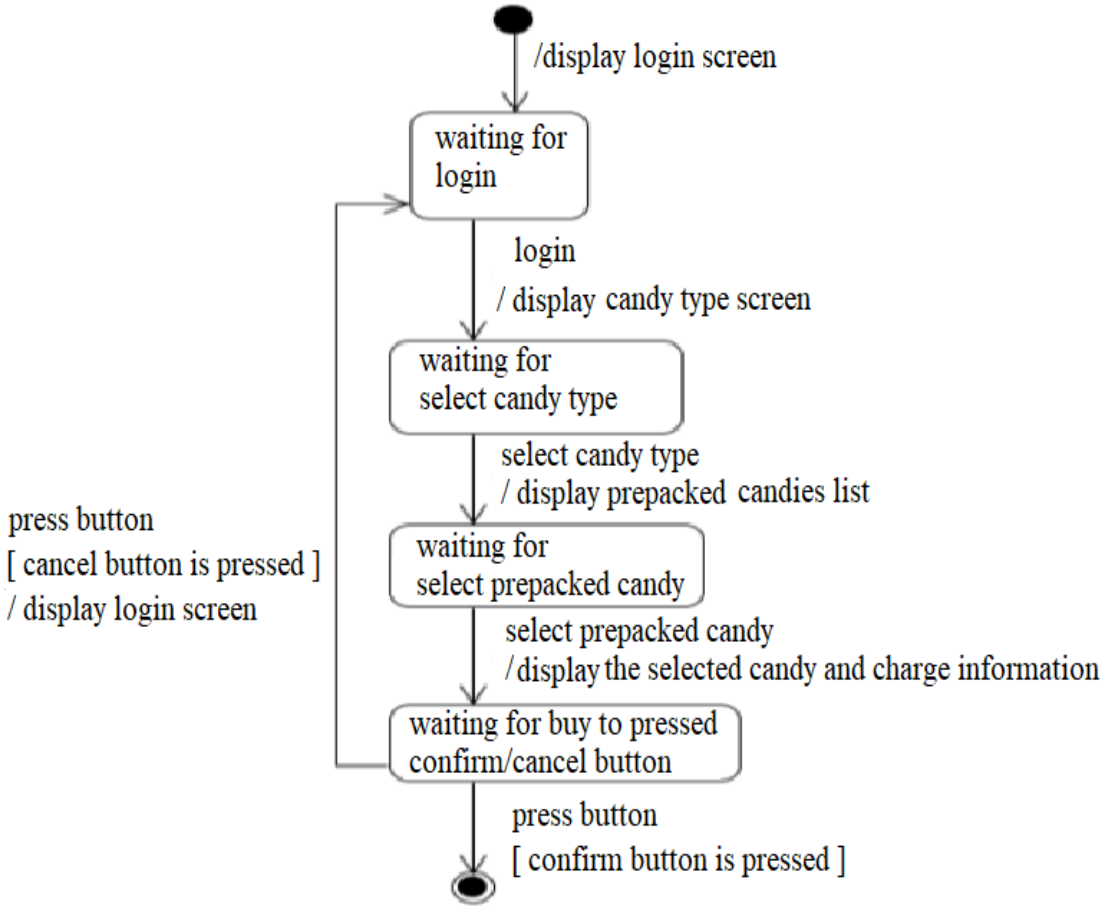
A3

(a)

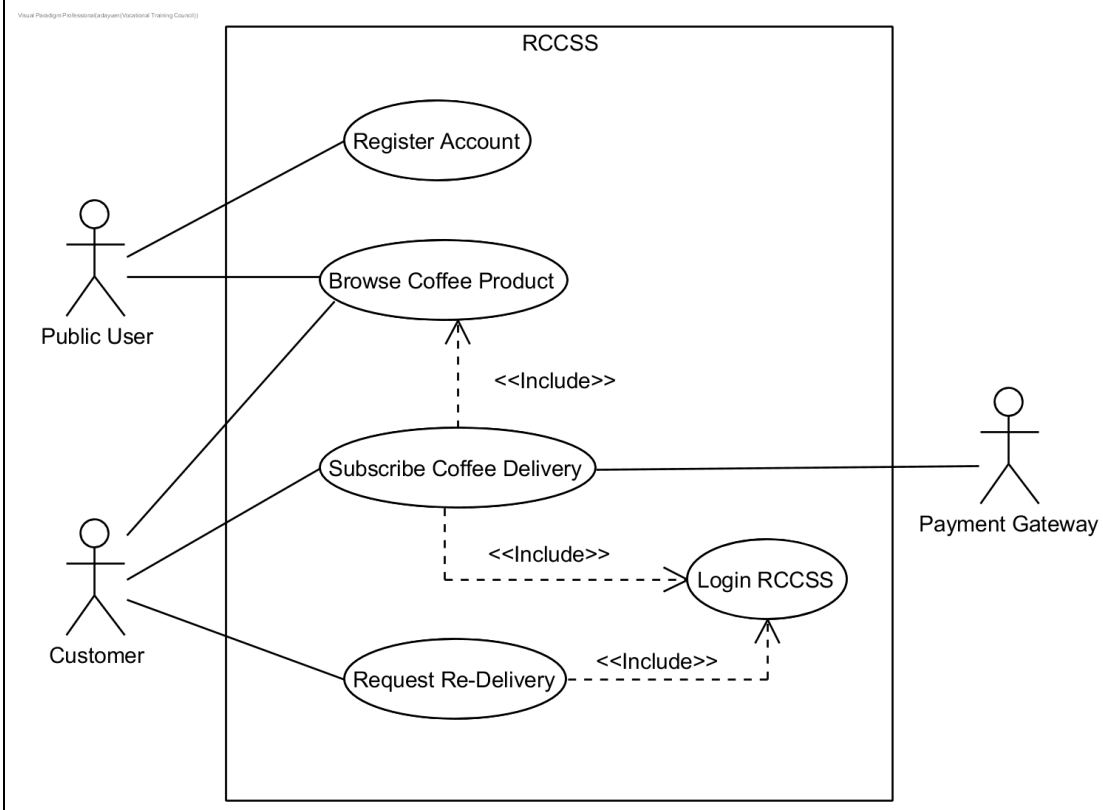

login
select candy type
select prepacked candy
press "Confirm"
or "Cancel" button
pressed cancel
button
pressed confirm
button

A3

(b)


/display login screen
waiting for
login
login
/ display candy type screen
waiting for
select candy type
select candy type
/ display prepacked candies list
waiting for
select prepacked candy
select prepacked candy
/display the selected candy and charge information
waiting for buy to pressed
confirm/cancel button
press button
[confirm button is pressed]
press button
[cancel button is pressed]
/ display login screen

B1


RCCSS
Register Account
Public User
Browse Coffee Product
<<Include>>
Subscribe Coffee Delivery
Payment Gateway
<<Include>>
Login RCCSS
Customer
Request Re-Delivery
<<Include>>

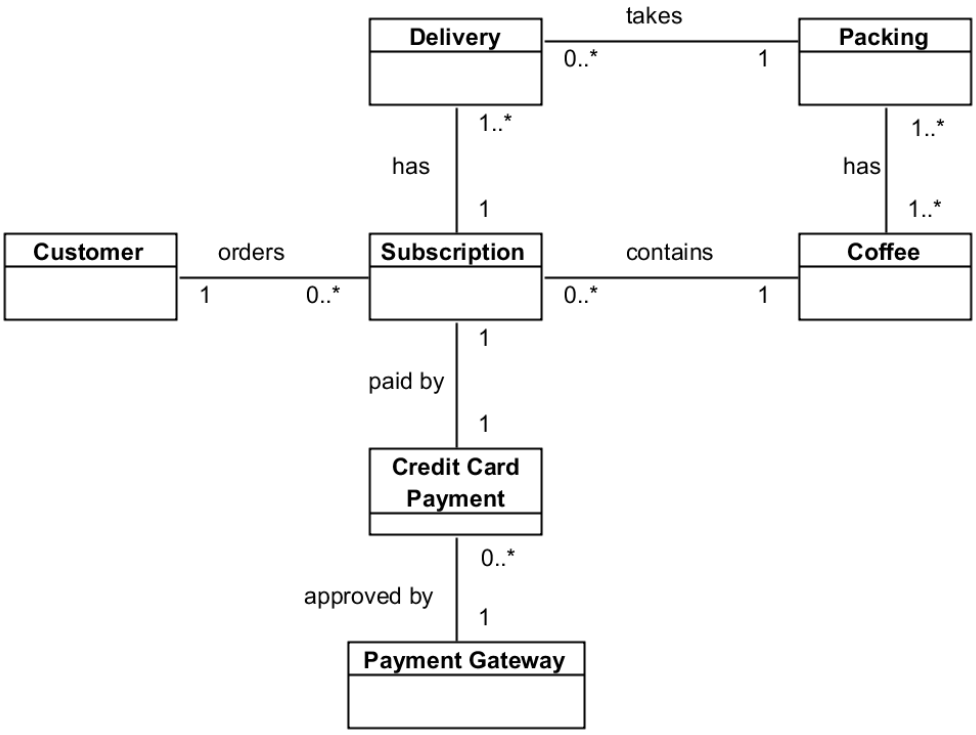
B2

(a)

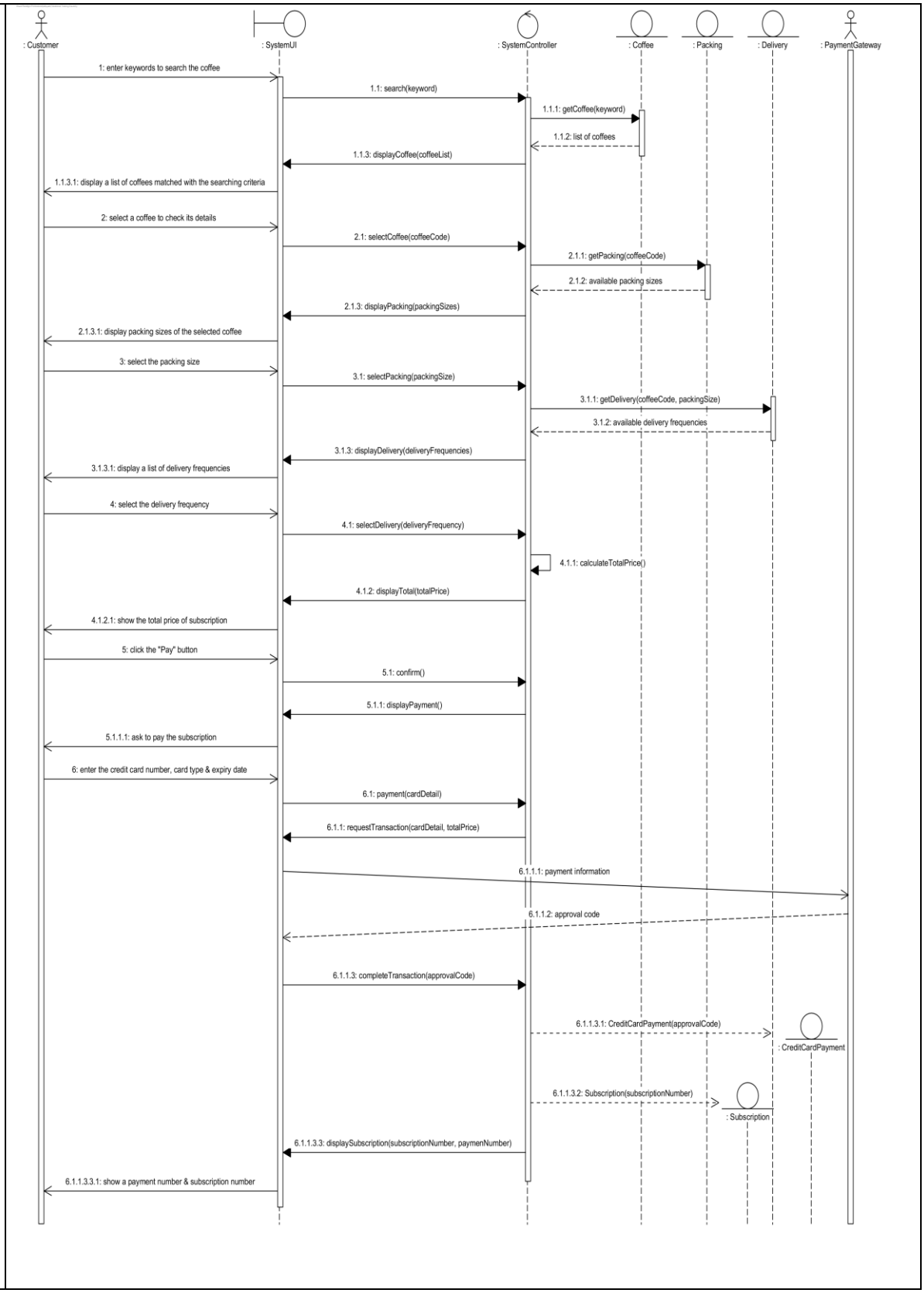
Candidate Class	Reason
Customer	Role Play
Subscription	Conceptual
Coffee	Tangible Thing
Packing	Conceptual Thing
Delivery	Event
Payment gateway	External System
Credit Card Payment/ Payment	Event

B2

(b)



B3



*****END OF Suggested Solution*****