I T P 4 9 0 9 - O O T - M A I N - 2 0 1 8 - M a y

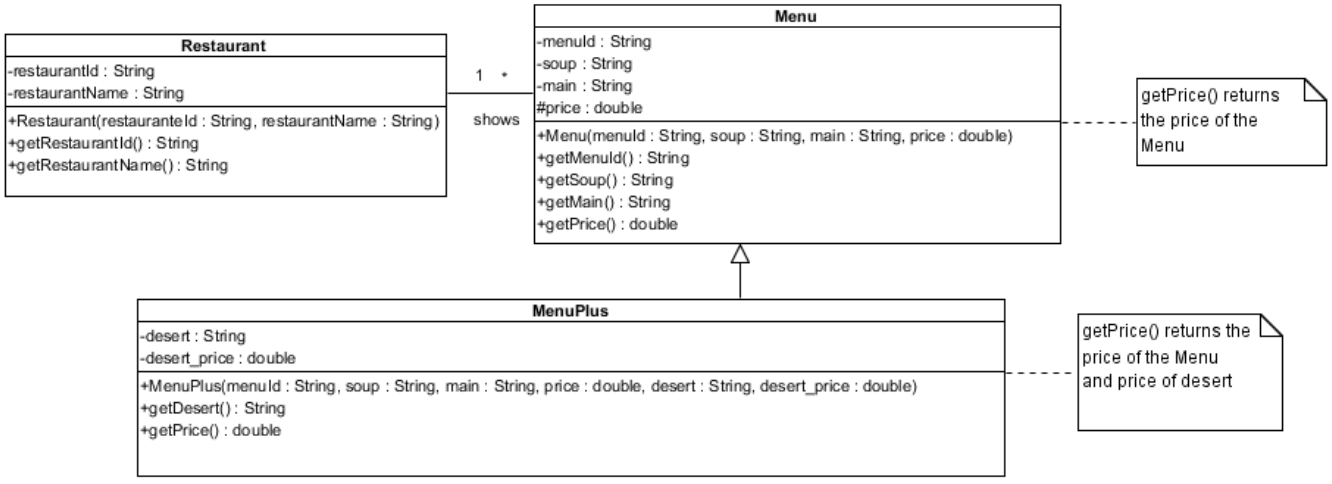
Section A (40 marks)
This section contains THREE questions. Answer ALL questions.

A1. A hospital provides different treatments for its patients. Patient is a person who stays in the hospital for treatment. Each patient has a hospital admission date. In the hospital, there are many wards. Each ward has a unique code and is gender specific. Each ward is shared by patients of the same gender who need similar kind of care. The doctors in the hospital are grouped into teams. Each team has a unique name and is led and managed by a senior doctor. The other members of the team are junior doctors. Each doctor has a unique ID for identification and can only be a member of one team. Each patient is stayed in a single ward and is under the care of a single team. A patient can only be treated by the doctors in the same team. On the other hand, a doctor can treat any number of patients in the team.

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.

[13 marks]

A2. The following class diagram and the test program are for listing menus in a restaurant. A menu has four basic pieces of information: ***menuId***, ***soup***, ***main*** and the ***price*** of the menu. In case a diner wants to have a desert to go with the menu, the diner will use the MenuPlus menu. The ***getPrice()*** method is to calculate the total amount a diner needs to pay for the meal.



QUESTION A2 CONTINUES ON THE NEXT PAGE

QUESTION A2 CONTINUES FROM THE PREVIOUS PAGE

```
public class Test {
  public static void main(String[] args) {
    Restaurant[] restaurants = new Restaurant[2];

    restaurants[0] = new Restaurant("R01", "Chan's Restaurant");
    restaurants[1] = new Restaurant("R02", "Delicious Restaurant");

    Menu[] menus = new Menu[3];

    menus[0] = new MenuPlus("SetA", "Mushroon Soup", "Fish", 40, "Cheese Cake",
          20);
    menus[1] = new MenuPlus("SetB", "Tomato Soup", "Steak", 50, "Chocolate
          Cake", 15);
    menus[2] = new Menu("SetB", "Beef Soup", "Pasta", 45);



    restaurants[0].addMenu(menus[0]);
    menus[0].setRestaurant(restaurants[0]);

    restaurants[0].addMenu(menus[2]);
    menus[2].setRestaurant(restaurants[0]);

    restaurants[1].addMenu(menus[1]);
    menus[1].setRestaurant(restaurants[1]);

    Enumeration<Menu> menuList = restaurants[0].getMenus();

    System.out.println ("Restaurant Name: " +
          restaurants[0].getRestaurantName());

    while (menuList.hasMoreElements()) {
          Menu menu = menuList.nextElement();

          System.out.println("Menu ID: " + menu.getMenuId() +
              "   Price: "  + menu.getPrice());
    }
  }
}
```

Based on the given class diagram and test program **Test.java**, implement the class ***Restaurant***, ***menu*** and ***MenuPlus*** in Java.

Your implementation should reflect the requirements in the class diagram and be able to run the **Test.java** program without error. [15 marks]

A3. Consider the steps of an applicant who uses a web-based system of a tertiary institution to apply for a programme admission. The applicant was assumed to be registered with the system already.

The system displays a login screen and asks the applicant to login the system. If the login is successful, the system displays a list of the departments in the institution. The applicant chooses a department for the application. The system will then display all the programmes in the department to the applicant. The applicant then chooses a programme to apply. The system then displays a summary of the programme chosen by the applicant along with two buttons "Confirm" and "Cancel". If the applicant presses the "Confirm" button, the application process will terminate and leave the system. If the "Cancel" button is pressed, the system will branch back to the login screen.

(a) Draw an activity diagram to model the procedure for an applicant to apply a programme in the institution. 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 an applicant to apply for a programme in the institution. 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.

Fit Fast is a fitness training company that provides full line of fitness training courses to customers through its 80 training centres. The company is now going to develop a Fit Fast Booking System (FFBS) to simplify its booking service of training courses.

A public user or a customer can use FFBS website to browse for the training courses provided by the company. There are two types of training courses that can be booked via FFBS, namely normal course and signature course. The normal type includes yoga, personal training, kickboxing, muscle training and aerobics. The signature type includes weekly assessment of physical fitness with an assessor and all courses covered by normal type.

Before a public user can book any training course, he/she has to register as a FFBS customer by providing his/her name, address, mobile phone number and email address in the FFBS website. Upon successful registration, a confirmation email with password will be sent to the registered customer.

A customer can book a training course through FFBS website by selecting a booking date and training centre for the training course. FFBS then displays a list of training courses available on the selected date and training centre. The customer then selects a training course. If the selected training course is a signature type, FFBS will ask the customer to select an available assessor for the training course. The customer selects an assessor. FFBS will ask the customer to login his/her account by entering email address and password. After successful login, FFBS shows the payment of the training course and asks the customer to pay the training course by credit card. The customer enters the credit card number, card type and expiry date. FFBS then forwards the payment amount together with the credit card details to an external payment gateway for approval. Upon successful approval of the credit card payment, the payment gateway sends back an approval code to FFBS. FFBS then shows a booking number and details of training course to the customer.

A customer can use FFBS to check his/her booking after logging in his/her account. The manager of FFBS training centre can use FFBS to browse all bookings after logging in his/her account. A manager can also use FFBS to create a training course by entering course type, available timeslots on particular dates, and the maximum number of customers allowed for booking.

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

B1. Draw a use case diagram for Fit Fast Booking System (FFBS). Show all required use cases, actors, communication links between actors and use cases, and <<include>>/<<extend>> relationships between use cases. [19 marks]

B2 (a) Perform a textual analysis on the problem statement to identify the candidate classes of the FFBS. List the candidate classes with the corresponding reasons/categories for your choices. [9 marks]

(b) Draw an initial class diagram to show the classes of the FFBS. Show the relationship among the classes in your answer. Give appropriate names to the association.

Note: You are NOT required to show the attributes/operations of the classes in your diagram and also the multiplicities of the associations. [8 marks]

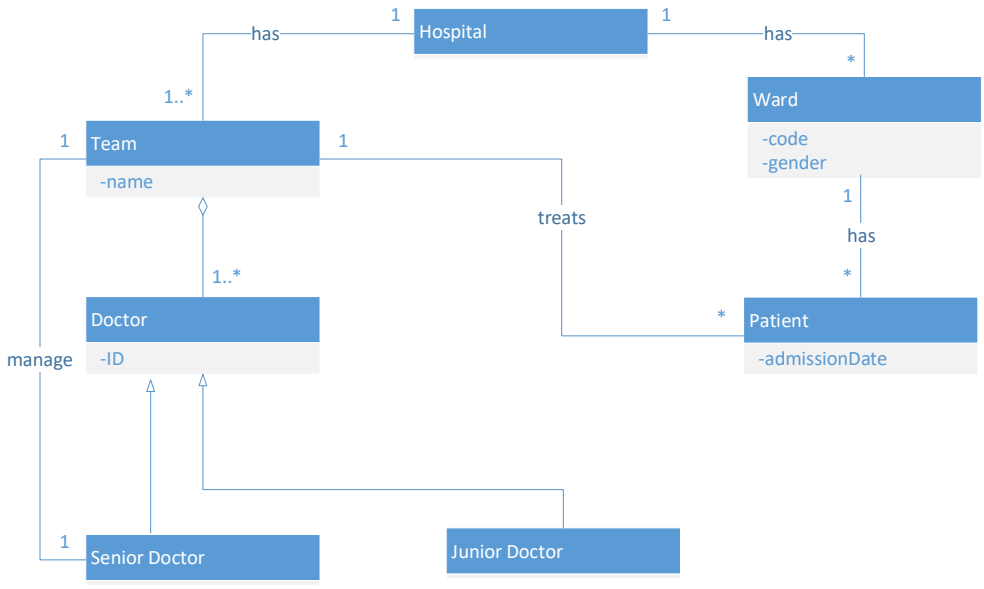
B3 Draw a three-tier (Model-View-Controller) sequence diagram of the Make Booking use case in the normal scenario for booking a signature course successfully.

Note: For simplicity, only ONE user interface (UI) object and ONE controller object are required for your design. [24 marks]

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

Suggested Solution

A1



A2

```
public class Restaurant {
    private ArrayList<Menu> menus = new ArrayList<Menu>(); //see Note

    private String restaurantId;
    private String restaurantName;

    public Restaurant(String restaurantId, String restaurantName) {
        this.restaurantId = restaurantId;
        this.restaurantName = restaurantName;
    }

    public String getRestaurantId() {  return this.restaurantId; }

    public String getRestaurantName() { return this.restaurantName;  }
//see Note
    public void addMenu(Menu menu) { menus.add(menu);   }
//see Note
    public Enumeration getMenus() { return Collections.enumeration(menus);}
}
```

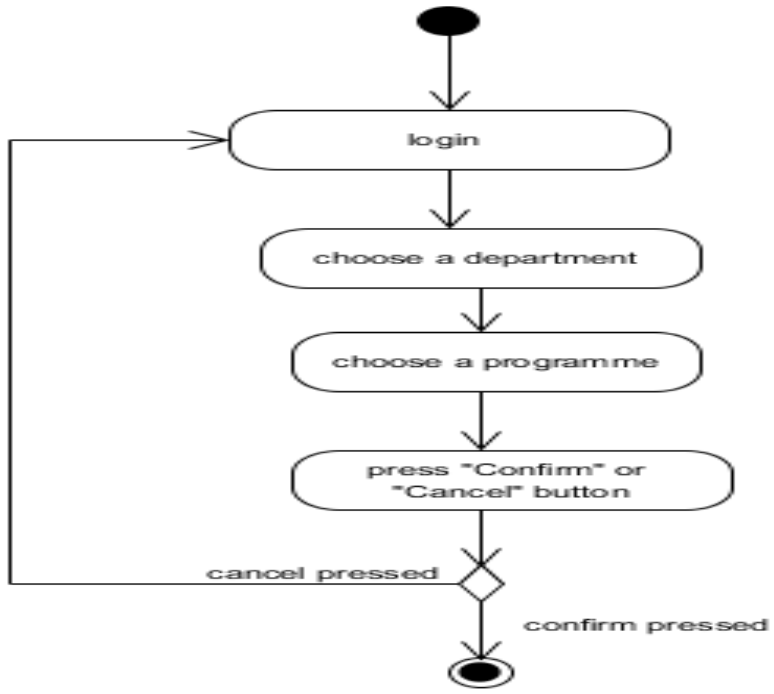
Note: one-to-many can be programmed using ArrayList or Vector or any other appropriate Java structure.

```
public class Menu {
    private Restaurant restaurant;

    private String menuId;
    private String soup;
    private String main;
    protected double price;

    public Menu (String menuId, String soup, String main, double price) {
        this.menuId = menuId;
        this.soup = soup;
        this.main = main;
        this.price = price;
    }


    public String getMenuId() {        return this.menuId;   }

    public String getSoup() {     return this.soup;}

    public String getMain() {     return this.main;}

    public void setRestaurant(Restaurant restaurant) {
        this.restaurant = restaurant;
    }

    public  double getPrice() {        return this.price;    }
}
```

```
public class MenuPlus extends Menu {
    private String desert;
    private double desert_price;

    public MenuPlus(String menuId, String soup, String main,
                    double amount, String desert, double desert_price) {
        super(menuId, soup, main, amount);

        this.desert = desert;
        this.desert_price = desert_price;
    }

    public String getDesert() {        return this.desert;   }

    @Override
    public double getPrice() { return super.price + this.desert_price;    }
}
```

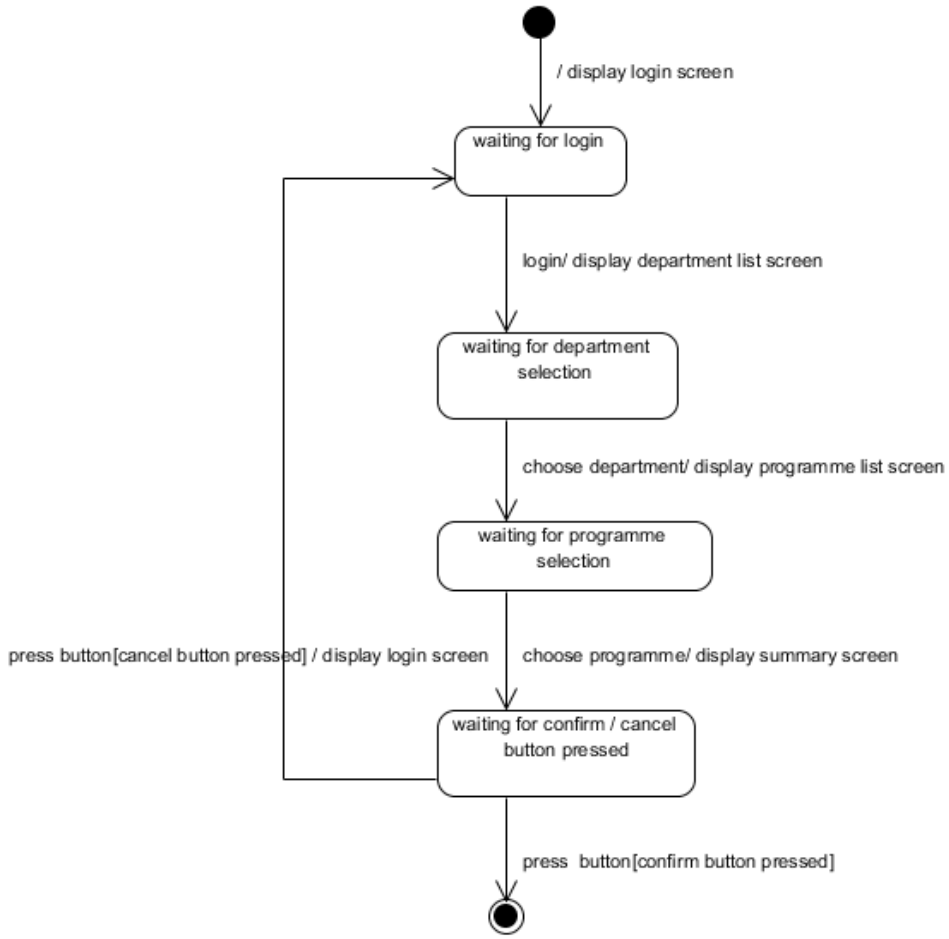
A3

(a)

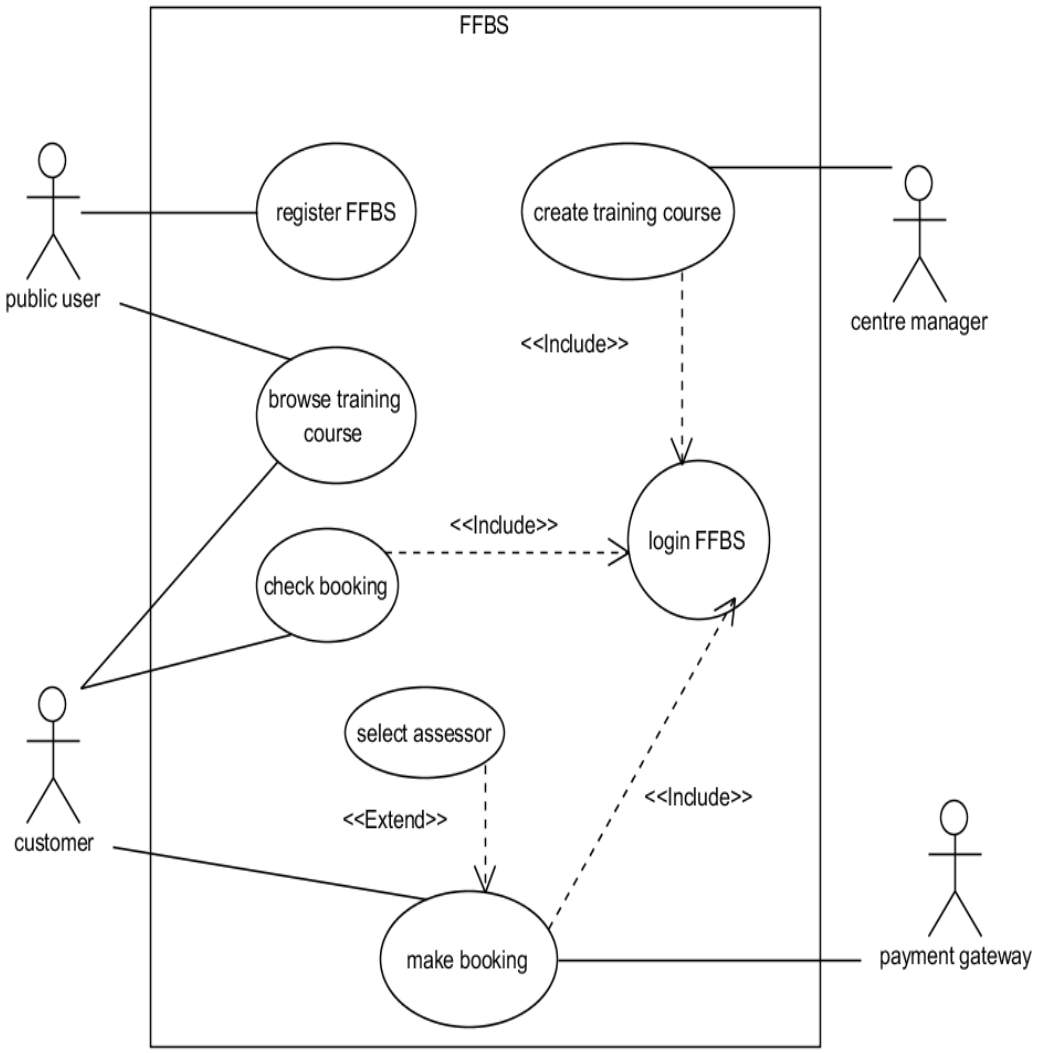

login
choose a department
choose a programme
press "Confirm" or "Cancel" button
cancel pressed
confirm pressed

A3

(b)


/ display login screen
waiting for login
login/ display department list screen
waiting for department selection
choose department/ display programme list screen
waiting for programme selection
press button[cancel button pressed] / display login screen
choose programme/ display summary screen
waiting for confirm / cancel button pressed
press button[confirm button pressed]

B1



Deduction for excess irrelevant use cases/communication links:

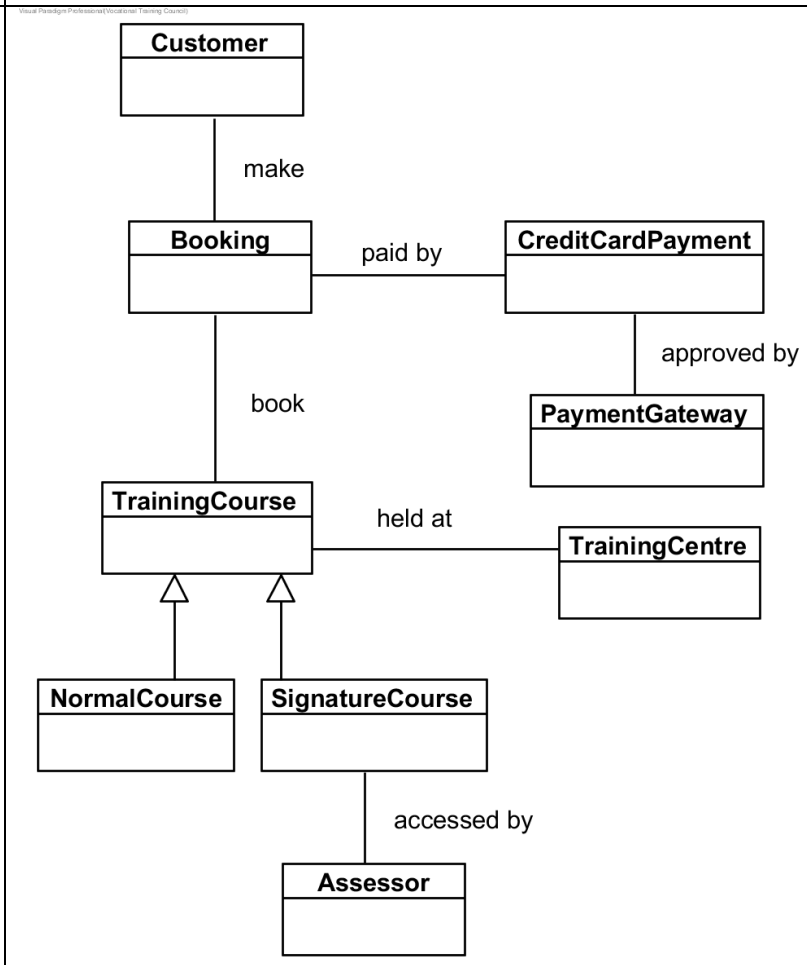
B2

(a)

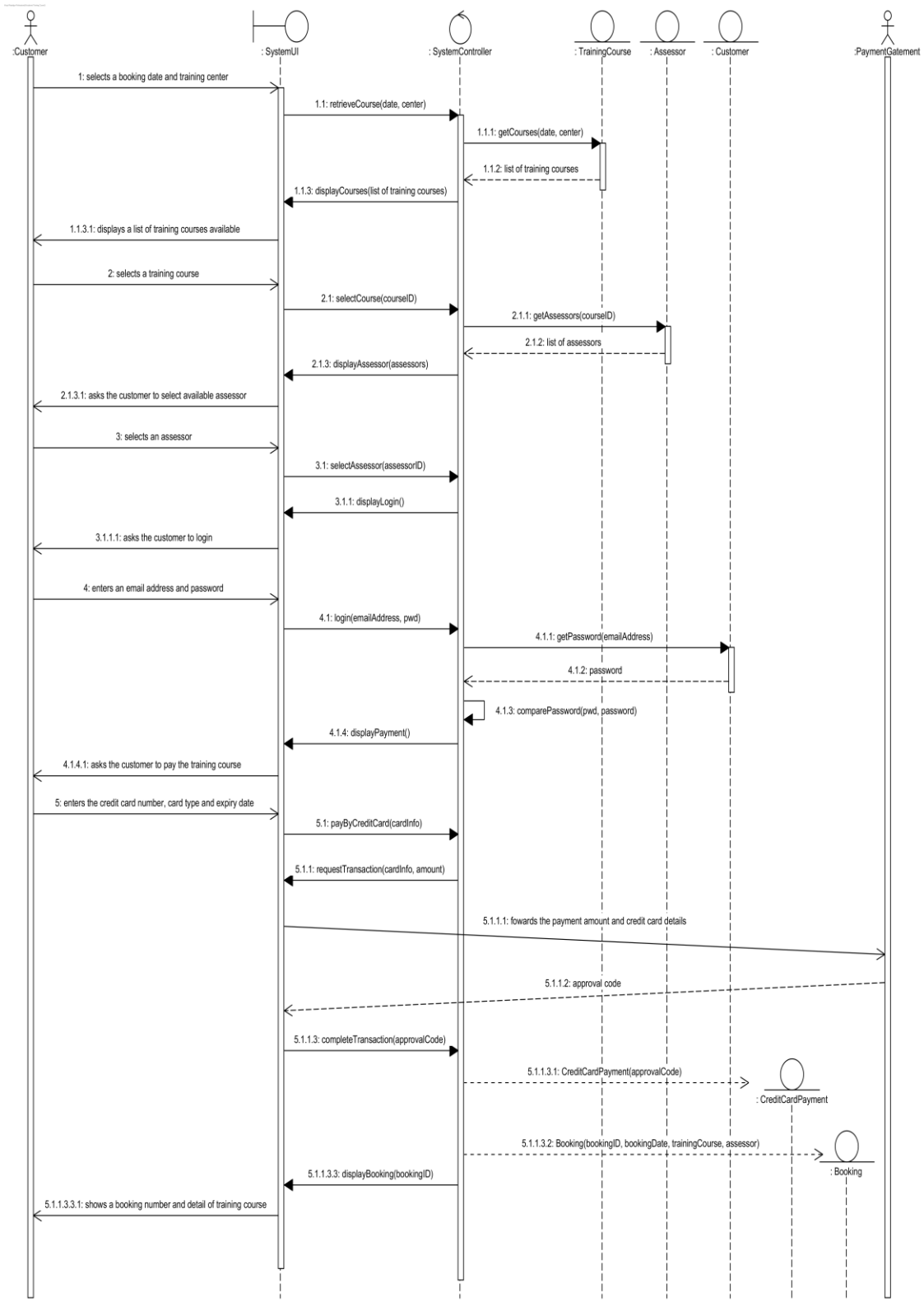
Candidate Class	Reason
Customer	Role Play
Booking	Conceptual / Event
Credit Card Payment	Conceptual / Event
Payment Gateway	External System
Training Course	Conceptual
Training Centre	Tangible
Normal Course	Conceptual
Signature Course	Conceptual
Assessor	Tangible

B2

(b)



B3



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