**Zehua Li**

**Assignment#1**

**Question#1:**

Class: Buyer, Amazon, Seller

**Buyer:**

Buyer can:1. Login in

2. Enter Payment method

3. Place Order &Check availability& Send Money

4. Receive Item

Data: Account, Password, BankCardNumber, Address, Camera,

Behavior:

LogIntoAmazonWebsite(){

Boolean.authenticated= Buyer.LoginWebsite(Account,password);

}

EnterPayment(){

If authenticated is valid;

Buyer.Response() = Buyer.EnterPaymentInformation(BankCardNumber,Addresss);

Else;

Account=Buyer.NewLoginWebsite(Account);

Password=Buyer.NewLoginWebsite(Password);

Buyer.LoginWebsite(Account,Password);

}

PlaceOrder(){

Boolean.authenticated= Buyer.EnterPaymentInformation(BankCardNumber,Addresss);

If authenticated is valid

RecieveAvaibilityFromSeller(Available);

If Available is true;

Buyer.PlaceOrder(Camera);

Buyer.SendMoney(Money);

Else

Buyer.RecieveAmazonResponse(“Not available”);

End;

Else

Address = Buyer.EnterPaymentInformation(NewAddress);

BankCardNumber = Buyer.EnterPaymentInformation(NewBankNumber);

}

ReceiveItem(){

BuyerReceiveItem(camera);

}

**Amazon**

Data: CameraAvailability, BuyerMoneyMoney, UserAccount, UserPassword

Behavior:

Amazon.ReceiveOrderFromBuyer() {

PleaseTypePaymentInformation(UserAccount,UserPassword);

}

Amazon.PlaceOrderToSeller(){

Amazon.RecieveInformation(UserAccount,UserPassword)

If (UserAccount,UserPassword) is true {

PlaceOrderToSeller(Camera)

Else

ResponseToSeller()=ThePaymentInformationIsIncorrectPleaseTypeAgain()

}

Amazon.CheckAvailability(){

If CameraAvailability is true;

NotifyBuyerThatItemIsAvailable();

Else

System.out.print(“Item is not available”);

End;

}

Amazon.NotifySellerSendItem(){

RecieveMoney(BuyerMoney);

NotifiySellerSendItem(Camera);

SendMoneyToSeller(BuyerMoney);

}

**Seller**

Data: Camera, available, BuyerMoney

Behavior:

RecieveNotifyFromAmazon(Camera) {

CheckItemAvailability(available);

If available is true;

SendMessageToAmazon(“ItemIsAvailable”)

Else;

SendMessageToAmazon(“ItemIsNotAvailable:);

End

}

Seller.SendItem(){

Buyer.SendItem(camera);

Buyer.RecieveMoney(BuyerMoney);

}

**Question#2**

Class: App, Buyers

Buyers can: 1. Receive News of local events from app

2. Choose to buy the tickets or not

3.Enter payment information

4.Get the ticket link

**App:**

Data: News, Option, PurchaseLink, CardName, NameAddress, BuyerMoney, TicketLink

Behavior:

ReceiveNews() {

News.SendNotificationToBuyers(News);

App.RecieveBuyersMessage(Option);

}

GetBuyerDescision(){

If yes == true;

App.SendLinkToBuyers(PurchasingLink);

If PurchaseLink == true;

App.AskBuyerEnterInfo(CardNumber,Name,Address);

Boolean authenticated=App.AskBuyerEnterInfo(CardNumber,Name,Address);

If aunthenticated is true;

App.RecieveMoney(BuyerMoney)

App.SendTicketToBuyer(TicketLink)

Else;

App.NotifyTheBuyerReenter()

NewInfo = App.AskForInfoAgain(CardNumber,Name,Address);

App.AskBuyerEnterInfo(CardNumber,Name,Address) = NewInfo;

Else

End;

}

**Buyer:**

Data: News, Option, Cardname, Address, TicketLink

Behavior:

ReceiveNews(){

Buyer.RecieveNotificationFromApp(News);

}

Buyer.MakeDecision(){

Buyer.DoOptions(Option);

}

DoPayment(){

If Option == true;

Buyer.EnterInfo(CardNumber, Name, Address);

Else;

End;

ReceiveTicket(){

Buyer.ReceiveTicket(TicketLink);

}

**Question#3**

Class: Patient, app, insurance provider, doctor

Patient can:1.Send an appointment request to app

2.Receive the answer from doctor.

**Patient:**

Data: Appointment, Answer

Behavior:

SendMessageToApp(){

Patient.SendMessageToApp(Appointment)

}

ReceiveAnswerFromDoctor(){

Patient.ReceiveMessage(Answer);

}

**App:**

Data:PatientAppointment, DoctorAnswer

Behavior:

ReceiveRequestFromPatient(){

App.ReceiveMessage(PatientAppointment)

}

SendMessageToProvider(){

App.SendMessageToProvider(PatientAppointment);

ReceiveMessageFromProvider(){

App.ReceiveMessage(DoctorAnswer);

}

SendMessageToPatient(){

App.SendMessageToPatient(DoctorAnswer);

}

InsuranceProvider:

Data: PatientAppointment, Answer

**Behavior:**

ReceiveAppointmentFromApp(){

Provider.ReceiveMessage(PatientAppointment);

}

SendAppointmentToDoctor(){

Provider.SendAppointmentToDoctor(PatientAppointment);

}

ReceiveAnswerFromDoctor(){

Provider.ReceiveAnswer(Answer);

}

SendAnswerToApp(){

Provider.SendAnswerToApp(Answer)；

}

**Doctor:**

Data: PatientAppointment,Answer

ReceiveAppointFromProvider(){

Doctor.ReceivePatientAppointment(PatientAppointment);

}

SendAnswerToProvider(){

Doctor.SendAnswerToProvider(Answer);

}

**Question#4;**

Class: JobFinder, Platform, Employer,Appointment

JobFinder can:1.Send Employer interested.

2.Recieve answer from Employer

3.Move on

**JobFinder:**

Data: Job, EmployerAnswer

SendInterestToPosition(Job) {

JobFinder.SendMessageToApp(Job)

}

RecieveAnswer(){

JobFInder.RecieveMessageFromEmployer(EmployerMessage);

}

MoveOn(){

If EmployerMessage == Yes;

JobFinder.MakeAppointWithEmployer(Appointment);

Else;

End;

**Platform:**

Data: JobFinderMessage, EmployerMessage,

RecieveTheRequestFromJobFinder(){

Platform.RecieveMesssageFromJobFinder(JobFinderMessage);

}

SendMessageToEmployer(){

Platform.SendMessageToEmployer(Messasge);

}

RecieveAnswerFromEmpolyer(){

PlatForm.RecieveAnswerFromEmpolyer(EmployerMessage);

Boolean auntheticated = RecieveAnswerFromEmpolyer(EmployerAnswer);

SendAnswer(){

SendMessageToJobFinder(EmployerAnswer);

}

**Employer:**

Data:Message, Decision,JobFinderMessage

Behavior:

RecieveMessageFromPlatform(){

Employer.RecieveMessageFromPlatform(Message);

}

SendMessageToPlatForm(){

Employer.SendMessageToPlatForm(Decision);

If Decision is Yes;

RecieveMessageFromJobfinder(JobFinderMessage);

Else;

End;

**Question#5**

Class: Buyer, App, Dominoser

Buyer can:1.Make an Order

2.Receive confirmation

3.Do payment

4.Get the status of pizza

5.Get pizza

**Buyer:**

Data: Pizza, Confirmation, status, PickupConfirmation, Money

Behavior:

MakeOrder(){

Buyer.MakeOrder(Pizza);

}

ReceiveConfirmation(){

Buyer.RecieveMessageFromApp(Confirmation);

}

DoPayment(){

If Confirmation == yes;

BuyerSendPayment(Money);

Else;

End;

GetStatus(){

Buyer.GetStatus(Pizza);

}

GetPizza(0{

Buyer.GetPizza(Pizza)’;

}

**App:**

Data: BuyerMessage, Confirmation, payment,status

Behavior:

RecieveMessageFromBuyer(){

App.RecieveMessageFromBuyer(BuyerMessage);

}

SendMessageToDominoser(){

App.SendMessageToDominoser(BuyerMessage);

}

RecieveMessageFromDominoser(){

App.RecieveMessageFromDominoser(Confirmation);

If Confirmation == Yes;

App.SendMessageToBuyer(Confirmation);

Else;

App.SendMessageToBuyer(“Not Available”);

}

ReceiveMoney(){

App.RecieveMoney(Payment);

App.SendStatus(Status);

}

**Dominoser:**

Data:

BuyerMessage, DominoserMessage,Status,Pizza, Payment.

Behavior:

ReceiveMessageFromApp(){

Dominosor.RecieveMessageFromApp(BuyerMessage);

}

SendMessageToApp(){

Dominosor.SendMessageToApp(DominoserMessage);

}

PlaceOrder(){

If Dominoser Message== yes;

Dominoser .RecieveMoney(Payment);

Dominoser .SendStatusToApp(Status);

Dominoser .SendPizza(Pizza);

Else;

End;