SaverLoader + save\_file : final {static} + SaverLoader() + save (f:File, s:Store) + load (f:File) : Store CustomerFactory EmployeeFactory serialize(o : OutputStream , s: Store) deserialize( i : InputStream) : Store + getInstance(): CustomerFactory + getInstance(): EmployeeFactory - CustomerFactory() - EmployeeFactory() Card + createCustomer(firstName: String, lastName: String): Customer +createEmployee(name: Name,priviledge: String): Employee + createCustomer(firstName: String, lastName: String, phoneNumber: String): Customer +createEmployee(name: Name, role: Role): Employee - cardNumber : String +createManager(name: String): Employee - cardExpirationDate : String +createCashier(name: String): Employee <<Enumeration>> - cv2 : String + createChef(name: String) : Employee OrderType + card() nstance 👚 - Inhouse + card(cardNumber : String ,cardExpirationDate : String , cv2 : String ) - TakeAway + getCardExpirationDate(): String - HomeDelivery + getCardNumber(): String + getCv2(): String + setCardExpirationDate(cardExpirationDate : String) <<Interface>> <<Interface>> paysWithCard + setCardNumber(cardNumber : String ) iCustomerFactory **IEmployeeFactory** 0...1 + setCv2(cv2 : String ) + updateCardInfo(cardNumber: String ,cardExpirationDate : String , cv2 : String ) + createCustomer(firstName: String, lastName: String): Customer +createEmployee(name: Name,priviledge: String): Employee + createCustomer(firstName: String, lastName: String, phoneNumber: String): Customer +createEmployee(name: Name, role: Role): Employee paysWithCard +createManager(name: String): Employee 0.. text +createCashier(name: String) : Employee + createChef(name: String) : Employee Order customerFactory <<Enumeration>> employeeFactory AddressType - orderID: int 0...1 - DateTime: Date Home orderDateTime: String Business dateFormat: DateFormat - isComplete: boolean - completedBy: Employee addressType - totalPrice: double 0...1 - amountReceived: double - amountReturned: double - isOrderedByCustomerWithMembership: boolean - isCardPayment: boolean - paysWithCard: Card - rewardPointGenerated: int Address - isPaid: boolean + Order(ID:int) - location : String Store deliveryAddress + addltemToOrder(newItem:Item) 0...1 addltemToOrderByAmount(newItem:Item ,Count:int ) <<Enumeration>> - storeName : String + address() getAmountReceived(): double - phoneNumber : String Role LoginInfo + address(address : String) getAmountReturned(): double + Manager + address(address : String , type: AddressType) - loginID : String + getCompletedBy(): Employee + Store() + Cashier + getAddressType() : AddressType Register - password : String + getCustomerWithMembership(): Customer + Store(name:String) + Chef + getLocation(): String + getDateFormat(): DateFormat + Store(name:String, location:String) - registerID: int + LoginInfo (ID : String , PW: String ) + canEditMenu() : boolean + setAddressType( addressType: AddressType) + getDateTime(): Date + Store(name:String,phone:String, location:String) + getLoginEmployee() : Employee + canAddKiosk() : boolean + getDeliveryAddress(): Address + setLocation(location : String) + addEmployee(name:String, loginID:String, password:String, option:int) : Employee setOfRegister + getLoginId(): String + canAddRegister(): boolean + getOrderDateTime(): String + addEmployee(name:String, loginID:String, password:String, role:Role): Employee + Register(ID:int) + getPassword() : String customerAddresses + canUseKiosk() : boolean + getOrderID(): int + addKiosk(e:Employee, id:int) : Kiosk + Register(ID:int, pizzaStore:Store) matchPassword(String password): boolean + canCompleteOrder() : boolean + getOrderItem(item: Item ): OrderItem + addMenuItem( e:Employee , menu:Menu , name:String , price:double, desc:String) : boolean + getPizzaStore(): Store + setLoginEmployee(loginEmployee : Employee + canCreateOrder(): boolean + getPaysWithCard() : Card + addMenuItem (e:Employee , name:String , price:double, desc:String) : boolean + getRegisterID():int + setLoginId(loginId: String ) getRewardPointGenerated(): int + addNewMember(firstName:String, LastName:String, customerPhoneNumber:String): Customer + setPizzaStore(pizzaStore:Store) + setPassword(password: String) + getSetOfItems(): Set<OrderItem> + addRegister(e:Employee, id:int): Register + setRegisterID(registerID: int ) + authorizeEditMenus(e:Employee ) : Set<Menu> + getTotalPrice(): double isRegisterOrder Customer + getTypeOfOrder(): OrderType + createOrderViaKiosk (kioskID: int ): Order mployeeLoginInfo 0...1 + getlsKioskOrder : Kiosk + createOrderViaRegister(e:Employee , orderID:int ) : Order - firstName: String + getlsRegisterOrder : Register + defineMenu(e:Employee , name:String , desc:String ) : Menu - lastName : String + isCardPayment(): boolean + deleteMenu (menu : Menu) : boolean - customerPhoneNumber : String + isComplete(): boolean setOfPlacedOrder + deleteMenultem (e: Employee , item : Item) - memberShipNumber : String + isOrderedByCustomerWithMembership(): boolean + editMenu(e:Employee , menu:Menu ) : Set<Item> - rewardsPoint∶int + isPaid(): boolean + getAddress(): Address + Customer(firstName : String , lastName : String) + makeCardPayment(amountReceived: double , paymentCard: Card ) : boolean + getEmployee(employeeID: String ): Employee + Customer(firstName : String , lastName, : String, phoneNumber : String ) Kiosk - makeCardPayment(amountReceived : double ,cardNumber: String ,cardExpirationDate: String , cv2: String ): boolean + getEmployeeSet(): Set<Employee> → customerWithMembership + addNewAddress ( address : Address ) : boolean worksForStore + getMember(customerID:String ) : Customer makeCashPayment(amountReceived: double ): boolean 0...1 kioskID: int 0...1 + addOrder (order : Order) - updateMembershipHoldingCustomer(member: Customer ): boolean + getMembers() : Set<Customer> + getCustomerAddress () : Set <Address> removeMultipleCountOfItemFromOrder(newItem: Item, Count: int): boolean + getPhoneNumber(): String setOfKiosk + getCustomerOrders() : Set <Order> setAmountReceived(amountReceived: double) + getMenu(menuName : String) : Menu + Kiosk(ID: int) + getCustomerPhoneNumber() : String setAmountReturned(amountReturned: setAmountReturned(double amountReturned)) + getItem(itemName : String) : Item + Kiosk(ID:int ,pizzaStore: Store) loginEmployee + getFirstName(): String setAmountReturned(amountReturned: double): void + getSetOfItems() : Set<Item> + getKioskID():int + getLastName(): String + setCompletedByEmployee(completedBy: Employee): boolean + getSetOfKiosk() : Set<Kiosk> isKioskOrder→ + setKioskID(kioskID:int) + getMemberShipNumber() : String + setDateTime(dateTime: Date) + getSetOfMenus(): Set<Menu> + getPizzaStore(): Store + getRewardsPoint(): int + getListOfPlacedOrder(): ArrayList<Order> + setOrderDateTime(orderDateTime: String) + setPizzaStore(pizzaStore:Store) Employee + removeAddress(address : Address ) : boolean + setIsKioskOrder( kiosk : Kiosk) + getSetOfRegister(): Set<Register> createOrder(orderID) + removeOrder(order : Order) : boolean + setIsRegisterOrder ( register : Register) + getStoreName(): String - employeeID : String 0..1 customerOrders , + setCustomerAddresses(customerAddresses : Set<Address> ) + setOrderID(orderID: int ) + initDefineMenu(e:Employee ): boolean - employeeName : String + setCustomerOrders(customerOrders : Set<Order> ) + updateMembershipHoldingCustomer(member: Customer ): boolean + loginAttempt(loginID:String , password:String ): Employee - employeeCounter : int + setCustomerPhoneNumber(customerPhoneNumber : String ) + updateToHomeDeliveryOrder(deliveryAddress: Address) + placeOrder(orderToBePlaced: Order): boolean completedB + Employee(name: String ,role: Role ) + setFirstName(firstName : String ) + updateToHomeDeliveryOrder(String Address) + removeMenuItems(e:Employee , menu:Menu, items:Set<Item> ) + getEmployeeID(): String + setLastName(lastName : String ) + updateToInHouseOrder() + setAddress(address:String) + getEmployeeLoginInfo(): LoginInfo + setMemberShipNumber(memberShipNumber: String ) + updateToTakeAwayOrder() + setMembers(members:Set<Customer> ) + getEmployeeName(): String + setRewardsPoint(rewardsPoint: int ) updateTotal() + setPhoneNumber(phoneNumber:String) + getRole(): Role calculateCustomerRewardPoints() - setTypeOfOrder(typeOfOrder: OrderType ) + setSetOfItems(setOfItems: Set<Item>) + getWorksForStore(): Store - generateCustomerID(): String - setTotalPrice(totalPrice: double) + setSetOfKiosk(setOfKiosk: Set<Kiosk>) + matchLoginInfo(loginID: String, password: String) initializeOrdersPointAddress() OrderItem setRewardPointGenerated(rewardPointGenerated: int ) + setListOfPlacedOrder(listOfPlacedOrder: Arraylist<Order> ) + setEmployeeID(employeeID : String ) employeeSet + setEmployeeLoginInfo(loginInfo: LoginInfo) setPaysWithCard(paysWithCard: Card) + setSetOfRegister(setOfRegister: Set<Register>) quantity : int setOrderedByCustomerWithMembership(isOrderedByCustomerWithMembership: boolean) + setSpecial(e:Employee, i:ltem, percentOff:double) + setEmployeeName(employeeName : String ) - subTotal : double - setIsPaid(isPaid: boolean) + setStoreName(storeName: String) + setRole(role : Role ) setDeliveryAddress(deliveryAddress: Address) - initializeSetElements() + OrderItem(orderedItem:Item, quantity:int) + setWorksForStore(worksForStore : Store ) - setDateFormat(dateFormat: DateFormat) - getKiosk(int) computeSubtotal():double - generateEmployeeID(): String getRegister(int) - setCustomerWithMembership(customerWithMembership: Customer) + decrementQuantity(): int - setCompletedBy(completedBy: Employee ) + decrementQuantitybyAmount(amount:int ): int setOfMenus setComplete(isComplete: boolean) + getItem(): Item setCardPayment(isCardPayment: boolean ) + getQuantity(): int makeOrderPayment(amountReceived: double) + getSubTotal(): double Menu createAsInHouseOrder() + incrementQuantity(): int createAsCashPayment() - menuDescription : String + incrementQuantitybyAmount(amount:int ): int calculateRewardPoints() - menuName : String + setQuantity(int) - setItem(Item) + Menu(name: String , desc: String ) - setSubTotal(double) + addltem(i: ltem ) : boolean + deleteltem(i : Item ): boolean + getMenuName(): String + getMenuDescription() : String + getMenuItems() : Set<Item> menultemsSet - itemName : String - itemPrice : Double - itemDescription : String - isSpecial : boolean - specialPercentageOfPrice : double

+ Item(name: String, price: double)

+ getSpecialPercentageOffPrice(): double

- setItemName(itemName: String ) - setItemPrice(itemPrice: double )

+ setSpecial(percentOff: double )

+ setItemDescription(itemDescription : String )

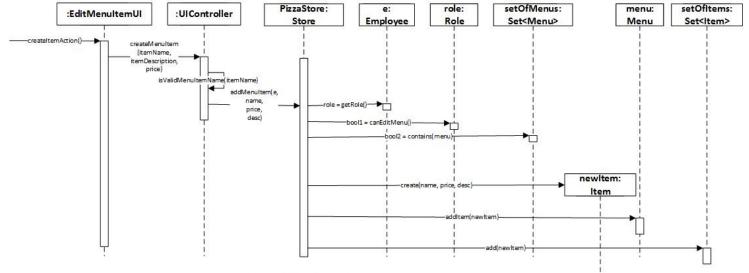
+ getCurrentPrice(): double + getItemBasePrice(): double + getItemDescription(): String + getItemName(): String

+ isSpecial() : boolean + removeSpecial()

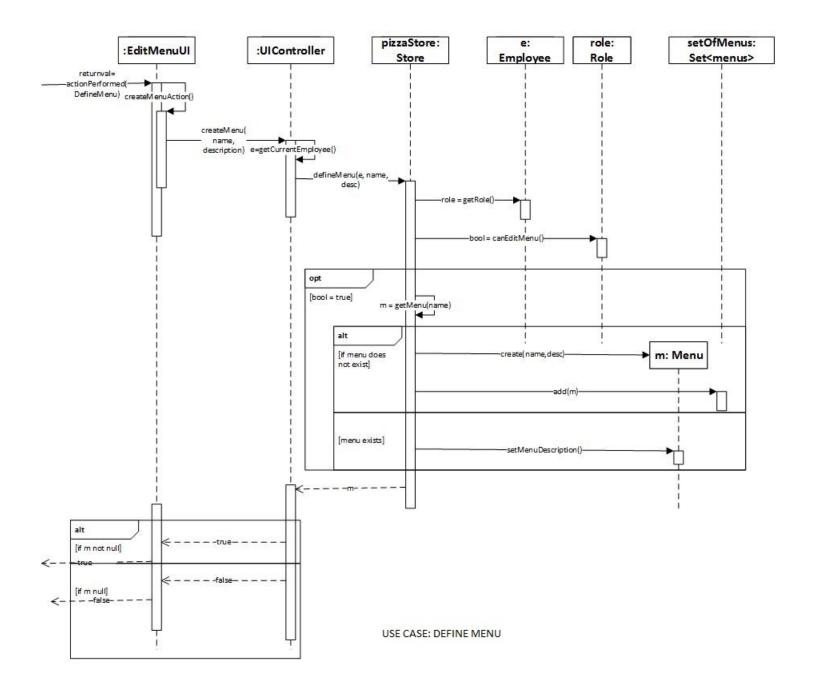
+ setSpecial()

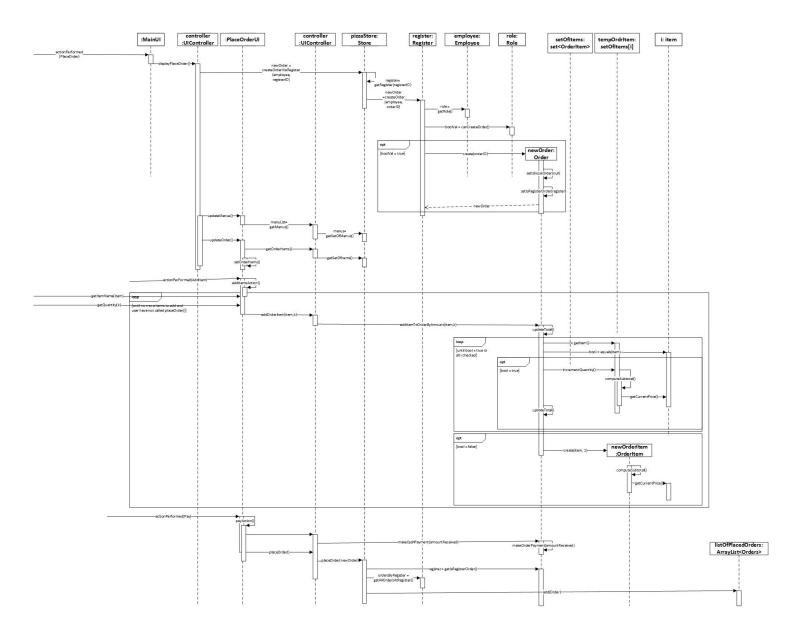
+ Item(name: String , price: double , itemDescription: String )

+ setSpecialPercentageOffPrice(specialPercentageOffPrice: double )



USE CASE: ADD MENU ITEM





USE CASE: Place order via regiter