#### **Group 0025 UML Diagram** Main <<Abstract>> Account - id : String - oldUsername : String + main(args: String[]): void oldPassword : String - username : String - password: String LoginSystem + Account(username : String, password : String) passwordLengthLimit : int + getId() : String invalidString : String[] + getOldUsername() : String - lp : LoginPresenter + getOldPassword() : String - ug : UserGateway + getUsername() : String - ag : AdminGateway + getPassword(): String - ig : ItemGateway + setId(accountId : String) : void mg : MeetingGateway + setId(): String - um : UserManager + setOldUsername(): void – adm : AdminManager + setOldPassword(): void - im : ItemManager + setUsername(username : String) : void – mm : MeetingManager + setPassword(password : String) : void - us : UserSystem + toString() : String - as : AdminSystem - gs : GuestSystem - input : Scanner **Admin** + LoginSystem() + run() : void + Admin(username : String, password : String) demold() : String loginMenu() : void + toString() : String - login() : String registerAccount() : void AdminManager - checkUsernameAvailability(): String - containsInvalidString(String anyString) : boolean adminList: ArrayList<Admin> + AdminManager(allAdmin : ArrayList<Admin>) LoginPresenter + getAdminList(): ArrayList<Admin> + addAdmin(newUsername: String, newPassword: String): - quit(): void + availableUsername(username: String): boolean + welcomeMessage() : void + incorrectPassword(username: String, pass: String): + options() : void + loginOptions() : void + logoutMessage() : void - adminByadminId(adminId: String): Admin - adminByUsername(username: String): Admin + adminLoginOptions(): void + userLoginOptions(): void + usernameByAdminId(userId String): String + adminIdByUsername(username String): String + closingMessage() : void + getUsernames(): ArrayList<String> + adminLoginMessage(): void + userLoginMessage() : void + changeUsername(adminId: String, username: String): void + enterUsernamePrompt() : void + changePassword(adminId: String, password: String); void + toString(): String + usernameNotFoundPrompt() : void - welcomeMessage(username : String) : void + enterPasswordPrompt() : void + incorrectPasswordPrompt(): void AdminSystem + invalidInputMessage() : void + welcomeNewUserMessage(username : String) : void ig: ItemGateway - ag: AdminGateway + userRegistrationMessage() : void - ug: UserGateway + newUsernamePrompt(): void - tg: ThresholdGateway + newPasswordPrompt(length : int) : void + invalidNewPasswordMessage() : void tdg: TradeGateway adm: AdminManager + proceedRegistrationPrompt(): void - um: UserManager + registrationResultMessage(isSuccessful : boolean) : UserManager) - im: ItemManager mg: MeetingGateway + usernameUnavailablePrompt() : void + changeUsernamePrompt() : void - mm: MeetingManager - tm: TradeManager + currentPasswordPrompt(): void - ap: AdminPresenter + newPasswordPrompt(): void - ip: ItemPresenter - up: UserPresenter tp: TradePresenter **AdminGateway** input: Scanner passwordLength: int adminInfo : HashMap<String,</li> - invalidString: String[] ArrayList<ArrayList<String>>> + AdminSystem(adm: AdminManager, um: UserManager) + AdminGateway() + run(adminId : String) : void + getAdminInfoCsv(): ArrayList<Admin> - manageProfileControls(adminId: String): void convertToAdmin(adminList: HashMap<String,</li> - manageUsersControls(): void ArrayList<String>>): ArrayList<Admin> - manageAllUsersControls(): void + updateAdminInfoCsv(adm: AdminManager) viewAllUsers(): void + getLoginInfo(): HashMap<String, ArrayList<String>> - undoControls(): void undoAddInventory(): void - undoRemoveInventory(): void undoAddWishlist(): void undoRemoveWishlist(): void void - isGoodSelection(selection: String, size: int) : boolean - isZero(selection: int): boolean - isZero(selection: String): boolean - undoUsername(): void - undoPassword(): void - userChoice(): String - tradeChoice(): String - newStatus(status: TradeStatus) : TradeStatus - currStatus(status: TradeStatus): String - userLastActed(userId: String, tradeId: String, status: TradeStatus): boolean - undoMeetingAction(userId: String, tradeId: String): void - undoConfirmed(userId: String, tradeId: String): void - undoTradeStatus(): void - helperUpdatePoints(tradeId: string): void resetPassword(): void + itemRemovalSuccess(): void - flagControls(): void + viewAllRequestedItems(ItemManager im): void - statusControls(): void + requestedItemMenu(): void - profilesByStatus(status: UserStatus): void - manageAdminControls(): void - manageItemsControls(): void - uncheckedOptions(itemId: String, userId: String): void uncheckedItems(): void - manageThresholdControls(): void - generatePassword(): String

- registerAdmin(): void

updateToCSV(): void

- checkUsernameAvailablity(): String

- containsInvalidString(anyString: String): boolean - changeUsername(adminId: String): void changePassword(adminId: String): void

- editThreshold(thresholdIndex: int, minValue): void

```
inventory : ArrayList<String>
 - wishlist : ArrayList<String>
 - status : UserStatus
 + User(username : String, password : String)
                                                                 String): boolean
 + getInventory(): ArrayList<String>
 + getWishlist(): ArrayList<String>
                                                                 - findUserByUserId(userId : String) : User
 + getStatus() : UserStatus
 + setInventory(inventory : ArrayList<String>) : void
 + setWishlist(wishlist : ArrayList<String>) : void
 + addToInventory(itemId : String) : boolean
                                                                 ArrayList<String>
 + removeFromInventory(itemId : String) : boolean
 + addToWishlist(itemId : String) : boolean
                                                                 + inventoryByUserId(userId : String) : ArrayList<String>
 + removeFromWishlist(itemId : String) : boolean
 + setStatus(status : UserStatus) : void
                                                                 boolean
 + toString(): String
                                                                 boolean
                      UserSystem
 - passwordLengthLimit: int
                                                                 ArrayList<String>) : void
 - isDemo : boolean
                                                                 + setWishlist(userId : String, wishlist :
 - ug: UserGateway
                                                                 ArrayList<String>) : void
 - thrG: ThresholdGateway
 - up: UserPresenter
 - um: UserManager
 - adm : AdminManager
 - im: ItemManager
 - tm: TradeManager
                                                                 boolean
 - input : Scanner
 + UserSystem(um : UserManager, adm : AdminManager,
 im: ItemManager)
 + run(userId : String) : void
                                                                 void
 - displayUserMenu(userId : String) : void
 - changeUsername(userId : String) : String
 - changePassword(userId : String) : String
 - viewProfile(userId : String) : void
                                                                  + checkUsername(username : String) : boolean
 - requestUnfreeze(userId : String) : void
                                                                  : boolean
                      Gateway
                                                                 + toStringByAccountId(userId : String) : String
+ gateList: ArrayList<String>
                                                                 + toString(): String
+ getSheetByFileName(sheetName: String): Scanner
+ getLoginInfo(): HashMap<String, ArrayList<String>>
                                                                  cancelString : String
                  AdminPresenter
- adm: AdminManager
                                                                  + UserPresenter()
- um: UserManager
                                                                 + getCancelString(): String
+ AdminPresenter(adm: AdminManager, um:
                                                                 + chooseOptionPrompt(): void
                                                                 + invalidInputMessage(): void
+ getCancelString(): String
+ chooseOptionPrompt(): void
+ invalidInputMessage(): void
                                                                 + userMenuOptions(): void
+ updateSuccessMessage(String thing): void
                                                                 + yourCurrentListMessage(isInventoryList : boolean) :
+ noUpdateMsg(): void
+ currentPasswordPrompt(): void
+ incorrectPasswordPrompt(): void
+ newPasswordPrompt(): void
+ invalidNewPasswordMessage(): void
+ changeUsernamePrompt(String username): void
+ adminMainMenu(): void
+ adminManageProfileMenu(): void
+ adminManageUserMenu(): void
+ flagMenu(): void
+ adminManageAdminMenu(): void
+ adminManageItemMenu(): void
+ adminManageThresholdMenu(): void
+ viewAdmin(): void
+ adminUsernameMessage(username: String): void
+ registrationResultMsg(isSuccessful: boolean): void
+ adminRegisterPrompt(): void
                                                                  userInfo : HashMap<String,</li>
 generatePassMsg(username: String, pass: String)
                                                                 ArrayList<ArrayList<String>>>
                                                                 + UserGateway()
+ usernameUnavailablePrompt(): void
+ proceedRegistrationPrompt(): void
+ viewUsers(): void
+ userProfile(userId: String): void
+ usernameNotFound(): void
+ frozenUsers(): void
+ requestedUsers(): void
+ flaggedUsers(): void
+ statusPrompt(String status): void
                                                                  input: Scanner
+ thresholdEditPrompt(String thresholdName): void
                                                                  gp: GuestPresenter
+ badThresholdMessage(int minValue): void
+ unacceptableThresholdMessage(int minValue): void
                                                                  ip:ItemPresenter
+ thresholdEditSuccessMessage(): void
                                                                  - im:ItemManager
+ viewAllItems(ItemManager im): void
                                                                  - um:UserManager
+ singleItemMenu(): void
```

# UserManager

- userIdToUser: HashMap<String, User>
- + UserManager(allUsers : ArrayList<User>)
- + getUserIdToUser(): HashMap<String, User> + addUser(newUsername : String, newPassword :
- findUserByUsername(username: String): User
- + usernameByUserId(userId : String) : String
- + userIdByUsername(username : String) : String + userIdsByStatus(status : UserStatus) :
- + statusByUserId(userId : String) : UserStatus
- + itemIdInInventory(userId : String, itemId : String) :
- + itemIdInWishlist(userId : String, itemId : String) :
- + wishlistByUserId(userId : String) : ArrayList<String>
- + setInventory(userId : String, inventory :

- + addToInventory(userId : String, itemId : String) :
- + removeFromInventory(userId : String, itemId : String) :
- + addToWishlist(userId : String, itemId : String) :
- + removeFromWishlist(userId : String, itemId : String) :
- + changeUsername(userId : String, username : String) :
- + changePassword(userId : String, password : String) :
- + changeStatus(userId : String, UserStatus status) :
- + getUsernames(): ArrayList<String>
- + checkPassword(username : String, password : String)
- + unfreezeRequest(userId : String) : boolean

- + findUserByItemInventory(itemId : String) : String
- + findUserByItemWishlist(itemId : String) : String

## UserPresenter

- + UserPresenter(cancelString : String)

- + incorrectPasswordPrompt(): void
- + mainMenuOptions(isFrozen : boolean) : void

- + changeUsernamePrompt(username: String): void
- + usernameUnavailablePrompt(): void
- + updateSuccessMessage(thing : String) : void
- + currentPasswordPrompt(): void
- + newPasswordPrompt(): void
- + invalidNewPasswordMessage(): void
- + frozenWarningMessage(): void
- + frozenReminderMessage(): void
- + unfreezeRequestSentMessage(): void
- + unfreezeRequestAlreadySentMessage(): void
- + displayProfile(userInfo : ArrayList<String>) : void

## UserGateway

- + getUserInfoCSV(): ArrayList<User>
- + convertToUser(userList: HashMap<String,
- ArrayList<ArrayList<String>>>): ArrayList<User>
- + updateUserInfoCsv(um : UserManager) : void
- + getLoginInfo(): HashMap<String, ArrayList<String>>

## GuestSystem

- + GuestSystem(im: ItemManager, um: UserManager)
- + run(): void
- guestItemMenu(): void
- browseltemsControls(): void
- quit(): void
- browseltems(im: ItemManager, um: UserManager):

## **Trade UML**

## DoActionTradeSystem

- tm:TradeManager
- im:ItemManager
- um:UserManager
- mm:MeetingManager - ms:MeetingSystem
- tp:TradePresenter = new TradePresenter()
- userId:String
- tg:TradeGateway = new TradeGateway()
- ig:ItemGateway = new ItemGateway()
- ug:UserGateway = new UserGateway()
- mg:MeetingGateway = new MeetingGateway()
- invalidStrings: String[] = new String[] {",", "/"}
- ~ DoActionTradeSystem(userId:String,
- tm:TradeManager, im:ItemManager, um:UserManager, mm:MeetingManager)
- run(listTrade: ArrayList<Trade>, isDemo:
- boolean):void
- helperSwitchConfirmOrMeeting(tradeIdInput: String):void
- helperSwithcAcceptRejectTrade(tradeIdInput: String):void
- eligibleToAcceptOrReject(tradeIdInput:
- String):boolean
- helperInitiateMeeting(tradeIdInput:String):void - helperPromptPlaceTime():Pair<String, LocalDate>
- helperPromptTime():LocalDate
- helperConfirmTrade(tradeIdInput: String):void
- eligibleToConfirmTrade(tradeIdInpu:
- String):boolean
- eligibleToViewMeeting(tradeIdInput:
- String):boolean
- helperCompleteTrade(tradeIdInput: String):void
- helperSwapInventories(tradeId:String):void
- promptTradeId(input:Scanner):String
- validTradeId(tradeId:String):boolean
- getUserId():String
- validIncompleteTrade(userId:String):boolean
- contansInvalidString(anyString: String):boolean
- updateToCsv():void

#### Trade

- tradeId:String
- status:TradeStatus duration:int
- meetingId:String
- requester:String
- receiver: String
- itemIds:List<String>
- dateCreated:LocalDate
- +TradeStatus:enum
- + Trade(userId1: String, userId2: String, itemIds: List<String>)
- + toString():String
- + getTradeId():String
- + getStatus(): TradeStatus
- + getDuration():int + getRequester():String
- + getReceiver():String + getItemIds():List<String>
- + getMeetingId():String
- + getDateCreated():LocalDate
- + setStatus(status:TradeStatus):void
- + setTradeId(tradeId:String):void
- setDateCreated(dateCreated:LocalDate):void
- + setDuration(duration:int);void
- + setRequester(userId1:String):void
- + setReceiver(userId2:String):void
- + setItemIds(itemIds:List<String>):void
- + setMeetingId(meetingId:String):void
- + isOneWay():boolean

## **TradeComparator**

+ compare(t1, t2):int

#### RequestTradeSystem

- isDemo: boolean
- userId:String
- tm:TradeManager
- im ItemManager
- um:UserManager
- ig:ItemGateway
- tg:TradeGateway
- ug:UserGateway
- tp:TradePresenter
- RequestTradeSystem(userId: String, tm: TradeManager, im: ItemManager, um: UserManager)
- run():void
- helperRequestTemPerItem(userId2:String, all:ArrayList<String>):void
- + suggestOneWay(userId: String): void
- + suggestTwoWay(userId: String): void
- validBorrowDiff(userId:String):boolean helperPermanentOrTemporary(userId2:String,
- all:ArrayList<String>, duration1:int):void
- helperRequestOneItem():ArrayList<String> -helperGetAvailableItemForUserId(userId: String)
- : ArrayList<String>
- helperRequestTwoItem():ArrayList<String> getUsernameInput(input:Scanne):String
- getItemInput(input:Scannr, userId2:String):
- hasItemToBeTraded(userId: String, itemId: String): boolean
- haveRequested(itemId : String) : boolean
- haveRequestedAllItem(itemList:
- ArrayList<String>): boolean
- getUserId():
- startWithWishList(userId: String): String
- startWithInventory(userId: String): String
- suggestItemToLend(userId: String, otherUserId: String): String
- suggestItemToBorrow(userId: String,
- otherUserId: String): String
- filterAlreadyRequested(itemList : ArrayList<String>, possibleRequesterId : String) :
- ArrayList<String>
- isBadInput(input: String, maxValue: int): boolean
- updateToCsv(): void

## TradeManager

- cancelledThreshold:int
- weeklyTradeLimit:int borrowDiff:int
- tradeIdToTrade: HashMap<String, Trade> = new HashMap<>()
- +TradeManager(listThresholds:List<Integer>, listTrade(List<Trade>)

+ requestTrade(userId1:String, userId2:String, itemIds:List<String>,

- + getCancelledThreshold():int
- + getWeeklyTradeLimit():int + getBorrowDiff():int
- + getTradeIdToTrade(): HashMap<String, Trade) + setCancelledThreshold(cancelledThreshold:int):void
- + setWeeklyTradeLimit(weeklyTradeLimit:int):void
- + setBorrowDiff(borrowDiff:int):void + addTrade(tradeId:String, trade:Trade):void
- duration:int):void + findTrade(tradeId:String):Trade
- + rejectTrade(tradeId:String):void
- + acceptTrade(tradeId:String):void
- + meetingConfiremd(tradeId:String):void + confirmTrade(tradeld:String, userId:String):void
- + completeTrade(tradeId:String):void
- + cancelTrade(tradeId:String):void
- + checkBorrowDiff(userId:String):boolean
- + tradesInOneWeek(userId:String):List<String> - helperTradesInOneWeek(trade:Trade, userId:String):boolean
- + checkTradeLimit(userId:String):boolean + checkIncompleteTrade(userId:String):boolean
- + getAllStatusTradeByUserId(userId:String,
- tradeStatus:TradeStatus):ArrayList<String>
- + statusConverter(tradeStatus:String): Trade.TradeStatus
- + getAllCompletedBorrowTradeByUserId(userId:Strin): ArrayList<String> + getAllCompletedLendTradeByUserId(userId:String): ArrayList<String>
- + getAllLendTradeIdByUserId(userId:String): ArrayList<String>
- + getAllBorrowTradeIdByUserId(userId:String): ArrayList<String> + getAllTradeIdByUserId(userId:Strng): ArrayList<String>
- + getAllTwoWayTradeIdByUserId(userId:String): ArrayList<String>
- + getAllOneWayTradeId():HashMap<String, Trade> + getAllTwoWayTradeId():HashMap<String, Trade>
- + getRecentBorrowedByUserId(userId:String, num:iny): ArrayList<String>
- + getRecentItemsLendByUserId(userId:String, num:int): ArrayList<String>
- + getRecentItemsInTwoWayByUserId(userId:String, num:int):ArrayList<String> + getTopTradersByUserId(userId:String, num:int):ArrayList<String>
- helperChoosingTopTraders(num:int, returnAl,:ArrayList<String> userToNumTrades:HashMap<String,Integer>):void
- helper Get Top Traders (user Id: String,al:HashMap<String,Integer>):HashMap<String, Integer>
- + createNewTrade(userId1:String, userId2:String, itemIds):ArrayList<String>:String
- + getTradeStatusByTradeId(tradeId:String):Trade.TradeStatus + getReceiverByTradeId(tradeId:String):String
- + getRequesterByTradeId(tradeId:String):String + getItemIdsByTradeId(tradeId:String):List<String>

+ setMeetingIdByTradeId(tradeId:Strin, meetingId:String):void

- + getDurationByTradeId(tradeId:String):int
- + getMeetingIdByTradeId(tradeId:String):String + createReverseTrade(tradeId:String):String

### TradeSystem

- tm:TradeManager um:UserManager
- im:ItemManager
- mm:MeetingManager
- tp:TradePresenter
- dts:DisplayTradeSystem
- rts:RequestTradeSystem
- dats:DoActionTradeSystem
- userId:String
- ig:ItemGateway
- tg:TradeGateway ug:UserGateway
- mg:MeetingGateway
- + TradeSystem(useId:String, um:UserManager)
- + run(isDemo: boolean):void
- helperBadSwitch(s:String):void
- helperGoodSwitch(s:String):void
- getUserId():String
- checkPreRequesting(): boolean
- suggestedTradesMenu(): void

#### DisplayTradeSystem

- tm:TradeManager im:ItemManager
- um:UserManager
- tp:TradePresenter
- up:UserPresenter
- ip:AdminPresenter
- userId:String
- dats:DoActionTradeSystem
- ~DisplayTradeSystem(userId:String, tm:TradeManager, im:ItemManager, um:
- UserManager, dats:DoActionTradeSystem) ~ run():void
- manageTradesMenu(): voidtradeByTypeMenu(): void
- ~ tradeByStatusMenu(): void
- helperDisplayTrade(tradeIds: ArrayList<String>):void
- helperDisplayAllTrade(): void
- helperDisplayRecent()
- displayRecentBorrowItems():void - displayRecentLendItems():void
- displayRecentTwoWayItems():void
- displayRecentItems(): void - displayTopTraders():void
- getUserId():String
- getIntInput():int
- isBadInput(input: String, maxValue: int) :

## **TradeGateway**

- + tradeInfo:HashMap<String,
- + TradeGateway()
- + getThresholdInfoCSV():List<Integer> + getTradeInfoCsv():ArrayList<Trade>
- ArrayList<ArrayList<String>>>):ArrayList<Trade> + updateTradeInfoCsv(tm:TradeManager):void

  - + TradePresenter() + getValidInput():ArrayList<String>
  - + getAllTypeOptions():void
  - + getUserNameNotification():void

  - + getUsernameTopTraders(usernameList:
  - + getManagementNotification():void
  - + getItemNotification():void
  - + getTradeNotification():void
  - + getTradeSelectionNotification(s1:String, s2:String):void
  - + getAcceptRejectNotification():void
  - + getConfirmNotification():void

  - + promptNumRecentItems(s:String):void

  - + violateBorrowDiff():void + violateIncompleteTrade():void
  - + invalidDate():void
  - + promptMeetingTime():void
  - getMeetingNotification(s:String):void

- + getItemNotification2():void
- + getRecentSelectionNotification(s1:String, s2:String
- + getSwitchConfirmOrMeetingNotification():void
- + successfulActionTrade(s:String):void
- :String):void + promptNumForTopTraders():void
- + noTradeActionAvailable():void
- + invalidSelection():void + invalidInput():void
- + promptMeetingPlace():void
- + getGoBackMenuNotification():void
- + getMeetingInfo(meeting:Meeting):void + meetingCancelledNotification():void

- ArrayList<ArrayList<String>>>
- + convertToTrade(tradeList:HashMap<String,

## **TradePresenter**

- validInput:ArrayList<String> = new ArrayList<>()
- + getSelectionMenu(s:String):void
- + getTypeTradeNotification(s:String):void
- + getItemLIstInfo(itemList: ArrayList<Item>):void + getTradeListInfo(tradeList:ArrayList<Trade>):void
- ArrayList<String>):void
- + getNotFound(s:String):void
- s3:String, s4:String):void
- + successfulRequestTradeMade(s:String):void
- + successfulInitiateMeeting(time:LocalDate, place
- + violateBorrowDiff():void
- + violateTradeLimit():void

#### **ItemPresenter**

- + iitemPresenter
- + ItemMenu():void
- + allItemsMenu():void
- + wishlistMenu():void
- + inventoryMenu():void
- + viewAllItems(im, um):void
- + viewAllTrades(im. um):void
- + viewWishlist(um, im, userId):void
- + viewInventory(um, im, userId):void
- + addItemName():void
- +addItemDescription():void
- + itemCreationSuccess: void
- + invalidItemName: void
- + invalidDecription: void
- + addItemPrompt: void
- + removeItemPrompt: void
- + alreadyWishlistPrompt: void
- + notInWishlistPrompt: void
- + notInInventoryPrompt: void
- + noItemsPrompt: void
- + noWishlistPrompt: void
- + addToWishlistMessage: void
- + removedFromWishlistPrompt: void
- + removeFromInventoryPrompt: void
- + displayItemProfile(itemInfo): void
- + displayItem(itemId, username, im): void

## **ItemStatus**

- UNCHECKED: constant
- AVAILABLE: constant
- UNAVAILABLE:
- constant
- REMOVED: constant

### Item UML

#### **ItemSvstem**

- -isDemo: boolean
- -im:ItemManager
- -um:UserManager
- -up:UserPresenter
- -ip:ItemPresenter -ig:ItemGateway
- ug:UserGateway
- -userId:String
- -invalidStrings: String[]=new String[] {".", "/"}
- -input: Scanner=new Scanner(System.in)
- +run(isDemo: boolean): void
- +allItemsMenu: void
- +addToInventorvMenu: void
- +inventoryMenu: void
- +wislistMenu: void
- +addToWishlist(itemId): void
- + removeFromWishlist(itemid): void
- +removeFromInventory(inventory, itemId): void
- + containsInvalidString(anvString): boolean
- + browseltems(im. um): void

#### Item

- name:String
- id:String
- description:String
- enum:ItemStatus
- + Item(inputName: String, inputDescription:
- String)
- + getName():String
- + getId:String
- + getDescription():String
- + getStatus:ItemStatus
- + setName:void
- + setId:void
- + setDescription:void
- + setStatus:void
- + toString():String

### **ItemManager**

- itemIdToItem:HashMap<String, Item> = new HashMap<>()
- + ItemManager(allItems: List<Item>)
- + getItemIdToItem: Hashmap<String, Item>
- + itemList: ArravList<Item>
- + createNewItem(inputName.
- inputDescription):String + itemByItemId(itemId):Item
- + removeBvItemId(itemId): boolean
- + deleteItem(itemId): void
- + reAddBvItemId(itemId): boolean
- + getToString(itemId): String
- + removeByItemId(itemId): boolean
- + getItemIdByItem(item): String
- + nameByItemId(itemId): String + descriptionBvItemId(itemId): String
- + getReguestedItems: String
- + itemIdsByStatus(status): ArrayList<String>
- + statusByltemId(itemId): ItemStatus
- + changeStatus(itemId. status): void
- + toString:String

#### **ItemGateway**

- -HashMap<String, ArrayList<String>> itemInfo
- +ItemGateway
- +getItemInfoCSV:Arraylist<Item>
- +convertToltem:ArrayList<Item>
- +updateItemInfoCSV(ItemManager)
- +convertToHash<ItemManager>:HashMap

# Meeting UML

