TradeManager

- listTrade: List<Trade>
- OPEN :String -CLOSE: String -CANCEL: String
- + TradeManager()
- + recentThreePartners(int): List<Integer>
- + getListTrade(): List<Trade>
- + addTrade(t: Trade): void
- + getTradeHistory(userId: int): List<Trade>
- + filterHistory(userId:int): List<Trade>
- +topThreePartners(userId:int): Lst<Integer>
- + recentThreeItem(userId): List<Integer> + getOpenTrade(userId: int):List<Trade>
- + getClosedTrade(userId:int): List<Trade>
- + getCancelledTrade(uerId: int): List<Trade>
- + getWaitTrade(userId: int): List<Trade>
- + checkInManager(tradeId: int): Boolean
- + getTradeById(tradeId: int): Trade
- + removeTrade(tradeId: int): Trade
- + comfirmComplete(tradeId: int): boolean
- + validateTrade(trade: Trade, borrower:User
- numLendBeforeBorrow: int):boolean
- +checkOneWayTrade(int):boolean
- +setUserStatus(int,int.String):void
- +getId(int,int):int
- +allPartners(int):List<Integer>
- +cancelTrade(int):void
- +openTrade(int, ItemManager): void

MessageManager

- listMessages: ArrayList<Message>
- + MessageManager()
- + getMessageFor(receiverId: int): ArrayList<Message>
- + createNewMessage(senderId: int, receivrId: int, message: String): void

FeedbackManager

- listReport: ArravList<Report>
- listReview: ArrayList<Review>
- + FeedbackManager()
- + setReview(userId1: int, userId2: int, point: int, reason: String): boolean
- + calculateRate(userId: int): double
- + updateReport(userId1: int, userId2: int, reason: String): boolean
- + getReviewById(userId: int): ArrayList<Review>
- + deleteReport(userId1: int, userId2: int): boolean
- + deleteReview(userId1: int, userId2: int): boolean
- + haveReport(receiverId: int, reporterId: int): boolean
- + haveReview(receiverId: int, reviewerId: int): boolean
- + haveReview(recieverId: int): boolean

<<interface>> Serializable

UML for Use Cases

ItemManager

- listItem: ArrayList<Item>
- listItemToAdd: ArrayList<Item>
- listDeletedItem: ArrayList<Item>
- + ItemManager()
- + getListItemTOAdd(): ArrayList<Item>
- + getAllItem(): ArrayList<Item>
- + getAllTradableItems(): ArrayList<Item>
- + getAllCategoryItem(): HashMap<Category, ArrayList<Integer>>
- + getNamebyId(itemId: int): String
- + getItembyId(itemId: int): Item
- + allTradableItemsFromOtherUser(userID: int): ArrayList<Item>
- + getItemsBvIds(listIds: ArravList<Integer>); ArravList<Item>
- + getRequestItemIdByName(itemName: String): Integer
- + getIDFromWaitingItem(item: Item): ArrayList<Integer>
- + getItemsIDs(items: ArrayList<Item>): ArrayList<Integer>
- + getTradableItems(items: ArrayList<Item>): ArrayList<Item>
- + getNotTradableItems(items: ArrayList<Item>): ArrayList<Item>
- + getAllCategoryItem(items: ArrayList<Item>): HashMap<Category, ArrayList<Integer>)
- + setTradable(listIds: ArrayList<Integer>, tradable: boolean): void
- + setTradable(item: Item, tradable: boolean): boolean
- + addItemToAllItemsList(item: Item): void
- + addItemToListDeletedItem(item: Item): void
- + requestAddItem(name: String, description: String, ownerId: int, category: Category): void
- + removeFromListItemToAdd(itemId: int): boolean
- + deleteItemFromListItemToAdd(targetItem: Item); void
- + getCategoryItem(category: Category): ArrayList<Integer>
- + getCategoryItem(category: Category, items: ArrayList<Item>): ArrayList<Integer>
- + getSortedCategory(category: HashMap<Category,
- ArrayList<Integer>>): ArrayList<Category>
- + searchItem(itemName: String): ArrayList<Integer>
- + getMatchItem(wishlist: ArrayList<Item>): ArrayList<Integer>
- getMostMatchItem(wishlist: ArrayList<Item>): ArrayList<Integer>
- + getSortedItemByFollows(um: UserManager): ArrayList<Item>
- getUsersByItemsFollowed(um: UserManager): HashMap<Integer, ArrayList<Integer>)
- getUsersByItem(um: UserManager, itemId: int): ArrayList<Integer>

MeetingManager

- -MeetingList: List<Meeting>
- systemDate: Calendar
- + MeetingManager()
- + getSystemDate(): Calendar
- + setSystemDate(year:int, month:int, day:int): void
- +getMeetingByUserId(userId: int): List<Meeting>
- +getUnCompleteMeetings(userId: int, tradeManager:
- TradeManager): List<Meeting>
- +getCompleteMeetings(userId: int): List<Meeting>
- +getUnConfirmTimePlace(userId: int, tradeManager: TradeManager): List<Meeting>
- + getUnConfirmMeeting(userId: int):List<Meeting>
- + getMeetingByIdNum(tradeId: int, numMeeting: int):
- + checkMeeting(meeting: Meeting): boolean
- + editTimePlace(meeting: Meeting, userId: int, year: int, month: int, day: int, hour: int, min: int, sec: int, maxMeetingTimePlaceEdits: int): boolean
- + confirmTimePlace(meeting: Meeting, userId: int, maxMeetingTimePlaceEdits: int): boolean
- + setMeetingConfirm(tradeManager:
- TradeManager, Meeting: meeting, userId: int): Boolean
- + setSecondMeeting(meeting: Meeting, meeting1: Meeting,
- userId: int, maxMeetingTimePlaceEdits: int): void
- +addMeeting(tradeld: int, userId1: int, userId2: int, meetingNum: int, tradeManager: TradeManager): Meeting
- + getOverTime(meeting: Meeting): boolean
- + getListOverTime(userId int): List<Meeting>
- + getEditOverThreshold(tradeManager: TradeManager, meeting: Meeting): String
- + sortByDate(meetings: List<Meeting>): List<Meeting>
- + undoConfirmTookPlace(tradeId: int, meetingNum: int, userId: int): boolean
- + toString(): String
- + meetingToTradeId(m: Meeting): int

ActionManager

- listOfAllActions: ArrayList<Action>
- listOfCurrentRevocableActions: ArrayList<Action>
- listOfDeletedRevocableActions: ArrayList<Action>
- mapOfUndoRequest: Map<Integer, Integer>
- + ActionManager()
- + getListOfAllActions(): ArrayList<Action>
- + getListOfCurrentRevocableActions():

ArrayList<Action>

- + getMapOfUndoRequest(): Map<Integer, Integer>
- + addActionToAllActionsList(actionOwnerID: int,
- userType: String, menuNumber: String, adjustableInt:
- int, adjustableStr: String): void + addActionToCurrentRevocableList(actionOwnerID:
- int, userType: String, menuNumber: String,
- adjustableInt: int, adjustableStr: String): void + addActionToDeletedRevocableList(action: Action):
- # helper maxAllID(): int

void

- # helper maxRevocableID(): int
- + deleteAction(actionID: int): void
- + getAllActionID(listOfAction: ArrayList<Action>): List<int>
- + findActionByID(actionID: int): Action
- + searchRevocableActionByID(userID: int):

ArrayList<Action>

- + checkRevocable(targetAction: Action): boolean
- + addUndoRequest(actionID: int, userID: int): boolean
- + deleteUndoRequest(actionID: int): boolean + checkUndoRequest(actionID: int): boolean