

Phase 1 UML Diagram Group 0147

Entities

<<abstract>>
Account

-username: String
-password: String

-Account(username: String, password: String)
+getUsername(): String
+getPassword(): String
+setPassword(newpassword: String): String
+equals(acclInfo: Account): boolean
+hashCode(): int
+toString(): String

User

-personalInventory: ArrayList<Item>
-wishlist: ArrayList<Item>
-messages: ArrayList<Message>
-frozenStatus: boolean
-tradePerWeek: int = 5
-threshold: int = 1
-limitOfIncompleteTrade: int = 5;

+User(username: String, password: String)
+getPersonalInventory(): ArrayList<Item>
+getWishlist(): ArrayList<Item>
+getThreshold(): int
+getLimitOfIncompleteTrade(): int
+getTradePerWeek(): int
+getFrozenStatus(): boolean
+getMessages(): ArrayList<Message>
+setPersonalInventory(personalInventory: ArrayList<Item>)
+setWishlist(wishlist: ArrayList<Item>)
+setFrozenStatus(status: boolean)
+setMessages(messages: ArrayList<Message>)
+setThreshold(newThreshold: int)
+setLimitOfIncompleteTrade(newLimit: int)
+setTradePerWeek(tradePerWeek: int)
+accountInfo(): String

Admin

+Admin(username: String,
password: String)

Item

-itemID: String
-description: String
-name: String
-ownerName: String

+Item(name: String, ownerName: String, description: String)
+setOwnerName
(newOwnerName: String)
+toString():String
+setItemID(String)
+getItemID():String
+getDescription(): String
+getName(): String
+getOwnerName(): String
+isEqual(item: Item): boolean

GlobalWishlist

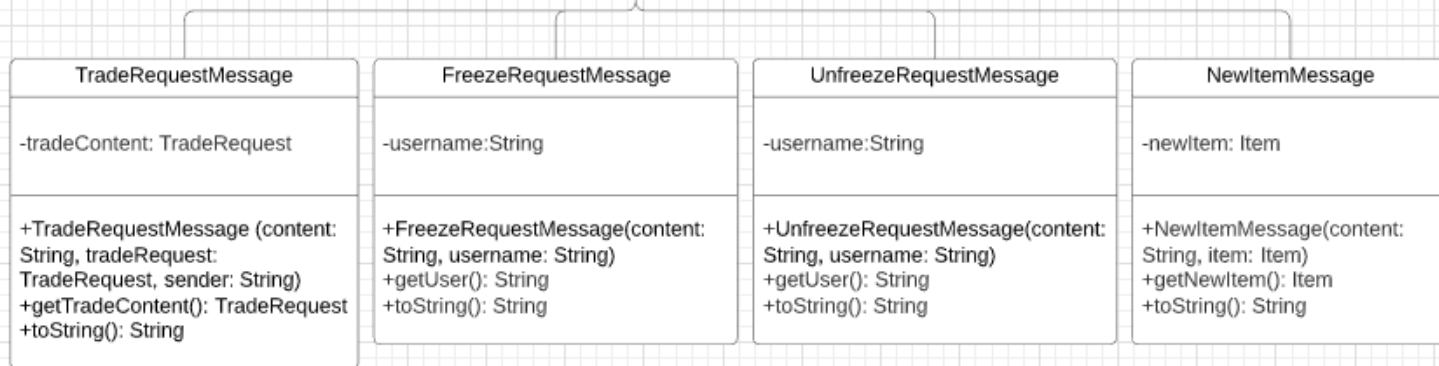
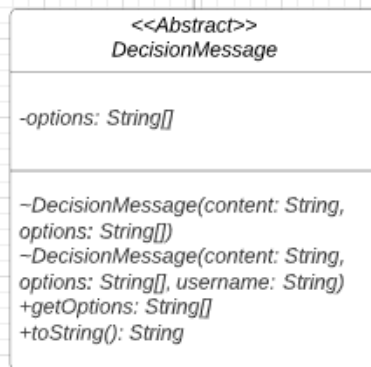
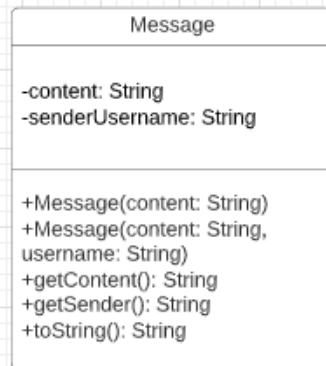
- wishMap: HashMap<String, ArrayList<String>>

+ GlobalWishlist()
+ addWish(itemid: String, userid: String): void
+ removeWish(itemid String, userid String): void
+ removeItem(itemid: String): void
+ isItemWanted(itemid: String): boolean
+ getFirstInterestedUser(itemid: String): String
+ getAllInterestedUsers(itemid: String): ArrayList<String>
+ toString(): String

GlobalInventory

-itemMap: HashMap<String, Item>
-itemCollection:
ArrayList<String>

+ GlobalInventory()
+ getItemMap():<String,Item>
+ setItemMap()
+ addItemToCollection
(itemID:String)
+ getIdCollection(): ArrayList<String>
+ getItem(itemID: String): Item
+ getItemByIndex(index: int):Item
+ removeItem(itemID: String)
+ getNumOfItem():int
+ containsKey(itemID: String)
:boolean
+ toString():String
+ addItem
(itemID: String, item: Item)
+isEmpty():boolean



TradeRequest

- userA: String
- userB: String
- itemA: ArrayList<Item>
- itemB: ArrayList<Item>
- perm: boolean
- date: LocalDateTime
- place: String
- confirmationA: boolean
- confirmationB: boolean
- numberOfEdit: int
- numberOfEdit: int

- + TradeRequest(userA: String, userB: String, itemb: ArrayList<Item>, perm: boolean, date: LocalDateTime, place: String)
- + TradeRequest(userA: String, userB: String, itemA: ArrayList<Item>, itemB: ArrayList<item>, perm: boolean, date: LocalDateTime, place: String)
- + getUserA(): String
- + getUserB(): String
- + getItemA(): ArrayList<Item>
- + getItemB(): ArrayList<Item>
- + isPerm(): boolean
- + getDate(): LocalDateTime
- + getPlace(): String
- + getNumberOfEditA(): int
- + getNumberOfEditB(): int
- + setDate(date: LocalDateTime)
- + setPlace(place: String)
- + setConfirmationA(confirmationA: boolean)
- + setConfirmationB(confirmationB: boolean)
- + setNumberOfEditA(numberOfEditA: int)
- + setNumberOfEditB(numberOfEditB: int)
- + getTradePartner(user: String)
- + toString(): String

Use Cases

userManager

- userManager: HashMap<String, User>

- + UserManager(allUsers: HashMap<String, User>)
- + login(username: String, password: String) : boolean
- + createNewUser(username: String, password: String, tradeHistory: HashMap<String, ArrayList<Trade>>) : boolean
- + changePassword(username: String, password: String) : void
- + isValidUser(username: String) : boolean
- + toString() : String
- + getCanTrade(user: String, borrowedTimes: int, lendTimes: int, numIncomplete: int, numTradesMadeThisWeek: int) : boolean
- + getCanTradesIgnoreBorrowLoans(username: String, numIncomplete: int, numTradesMadeThisWeek: int) : boolean
- + getUserWishlist(user: String) : ArrayList<Item>
- + getUserInventory(user: String) : ArrayList<Item>
- + getUserMessages(username: String) : ArrayList<Message>
- + setUserMessages(username: String, message: ArrayList<Message>) : void
- + getUserIncompleteTrades(username: String) : int
- + getTradesPerWeekForUser(username: String) : int
- + getUserInfo(username: String) : String
- + getUserFrozenStatus(username: String) : boolean
- + getUserThreshold(username: String) : int
- + addUserMessage(username: String, message: Message) : void
- + removeUserMessage(username: String, message: Message) : void
- + removeItemFromUserInventory(user: String, itemID: String) : void
- + removeItemFromUserWishlist(user: String, itemID: String) : void
- + removeFromMultipleUsersWishlists(users: ArrayList<String>, itemID: String) : void
- + addItemToUserInventory(item: Item, username: String) : void
- + addItemToWishlist(username: String, item: Item) : void
- + createNewItem(username: String, itemName: String, description: String) : NewItemMessage
- + createUserMessage(username: String, content: String) : void
- + createAndAddNewTradeRequestMessage(recipient: String, messageContent: String, tradeRequest: TradeRequest, senderUsername: String) : void
- + freezeUserAccount(username: String, freezeStatus: boolean) : void
- + freezeUserAccount(username: String) : void
- + setWeeklyTrades(newTradesPerWeek: int) : void
- + setWeeklyTradesForOneUser(username: String, newTradesPerWeek: int) : void
- + removeUser(username: String) : void
- + setLimitOfIncompleteTrades(newLimit: int) : void
- + setLimitOfIncompleteTradesForOneUser(username: String, newLimit: int) : void
- + setNewThreshold(newThreshold: int) : void
- + setNewThresholdForOneUser(username: String, newThreshold: int) : void

TradeManager

- tradeHistory: HashMap<String, ArrayList<Trade>>

+ TradeManager(tradeHistory: HashMap<String, ArrayList<Trade>>)
+ getTradeHistory(username: String): ArrayList<Trade>
+ getRecentCompletedTrade(username: String): Trade[]
+ getBorrowedTimes(username: String): int
+ getLendTimes(username: String): int
+ getFrequentTradingPartners(username: String): String[]
+ addTrade(trade: Trade)
+ tradesToConfirm(username: String): ArrayList<Trade>
+ getIncompleteTimes(username: String): int
+ numberOfTradesCreatedThisWeek(username: String): int
+ setConfirm(username: String, trade: Trade, status: boolean)

GlobalWishlistManager

- wishMap: HashMap<String, ArrayList<String>>

+ GlobalWishlistManager()
+ addWish(itemid: String, userid: String): void
+ removeWish(itemid: String, userid: String): void
+ removeItem(itemid: String): void
+ isItemWanted(itemid: String): boolean
+ userWhoWants(allItems: ArrayList<Item>): ArrayList<String>
+ getAllInterestedUsers(itemid: String): ArrayList<String>

TradeRequestManager

- t: TradeRequest
- canEditA: boolean
- canEditB: boolean

+ setDate(user: String, date: LocalDateTime)
+ setPlace(user: String, place: String)
+ setDateAndPlace(user: String, date: LocalDateTime, place: String)
+ setConfirmation(user: String): Trade
+ getTradeRequest(): TradeRequest
+ canEdit(user: String): boolean
+ getTrade(): Trade

GlobalInventoryManager

-gl: GlobalInventory

+GlobalInventoryManager(gl: GlobalInventory)
+ getItemFromGi(itemID: String):Item
+addItemToHashMap(item: Item)
+removeItem (itemID: String)
+generatePage (pageNumber: int): ArrayList<Item>
+generatePageNumber():int
+contains(item: Item): boolean
+hasNoItem():boolean

AdminManager

- adminHashMap: HashMap <String, Admin>
- adminMessagesArrayList: ArrayList <Message>

+ AdminManager (adminHashMap: Hashmap<String, Admin>, adminMessagesArrayList: ArrayList<Message>, userHashMap: HashMap<String, User>)
+ addAdmin (toAdd: Admin): HashMap<String, Admin>
+ addAdmin (username: String, password: String): void
+ getAdminMessagesArrayList(): ArrayList<Message>
+ setAdminMessagesArrayList(adminMessagesArrayList: ArrayList<Message>): void
+ addNewPassWord (password1: String, password2: String, admin: Admin): boolean

Presenters

AdminAccountPresenter

-admin: Admin

```
+AdminAccountPresenter(admin: Admin)
+printMainMenu()
+askForPassword()
+askToConfirmPassword()
+passwordChanged()
+newAdminUserName()
+newAdminPassword()
+failToCreateNewAdmin()
+exitMenu()
```

AdminBrowsingUsersPresenter

```
+ AdminBrowsingUsersPresenter()
+ enterUser(): void
+ invalidUser(): void
+ infoUser(info: String): void
+ thresholdUser(): void
+ tradelimitUser(): void
+ incomptradeUser(): void
+ successUser(): void
+ thresholdsuccessUser(): void
+ freezingUser(): void
+ errorUser(): void
+ invalidUser(): void
+ invalidoptionUser(): void
```

AdminMenu

-admin: Admin

```
+printMainOption()
+exitPresenter()
+goIntoMessageInbox()
```

MainMenuPresenter

```
+ printExit(): void
+ printMenuPrompt(): void
+ printLoginPrompt1(): void
+ printLoginPromptNewUsername(): void
+ printLoginPrompt2(): void
+ inputError(): void
+ savingError(): void
+ wrongLogin(): void
+ readError(): void
+ takenUsername(): void
+ usernameTooShort(): void
+ successfulAccountCreation(): void
```

GlobalInventoryPresenter

- properties: ArrayList<String>
+ gim: GlobalInventoryManager
- current: int

+ GlobalInventoryPresenter(gin: GlobalInventoryManager)
+ hasNext(): boolean
+ next(): String
+ printpage(page:int)
+ emptyPage()
+ addToWishlistandTradeRequest(item: Item)
+ addToWishlist(item: Item)
+ error()
+ addedToWishlist(item: Item)
+ invalid()
+ ownItem()
+ alreadyHave()
+ FrozenAcc

MessageReplyMenu

+MessageReplyMenu()
+printMenuPrompt(messageSize: int)
+printNoMessages()
+printExit()
+printContentMessagePrompt(m: Message)
+printDecisionMessagePrompt(m: DecisionMessage)
+printEditTradeRequestPrompt(t: TradeRequest)
+printErrorOccurred()
+printInvalidInput()
+tradeRequestWarning()
+tradeRequestCancel()
+changeDatePrompt(oldDate: LocalDateTime)
+changePlacePrompt(oldPlace: String)
+wrongFormat()
+printCannotTradePrompt()
+success()
+wrongDate()

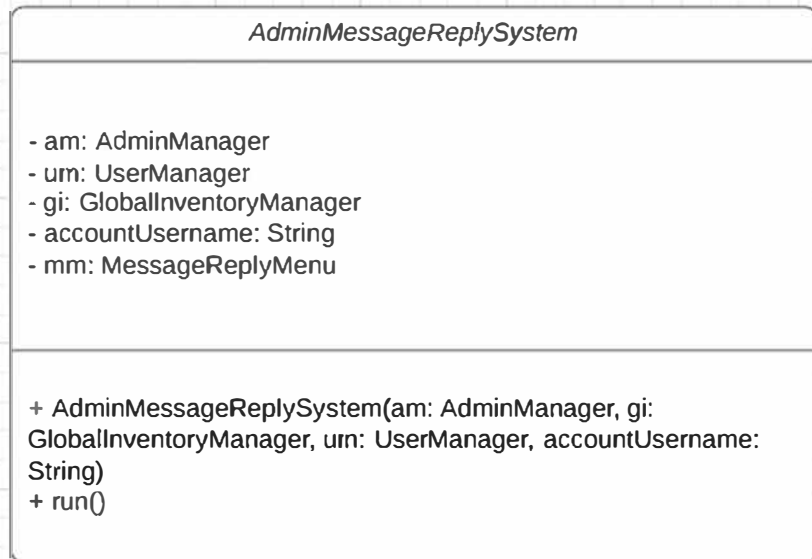
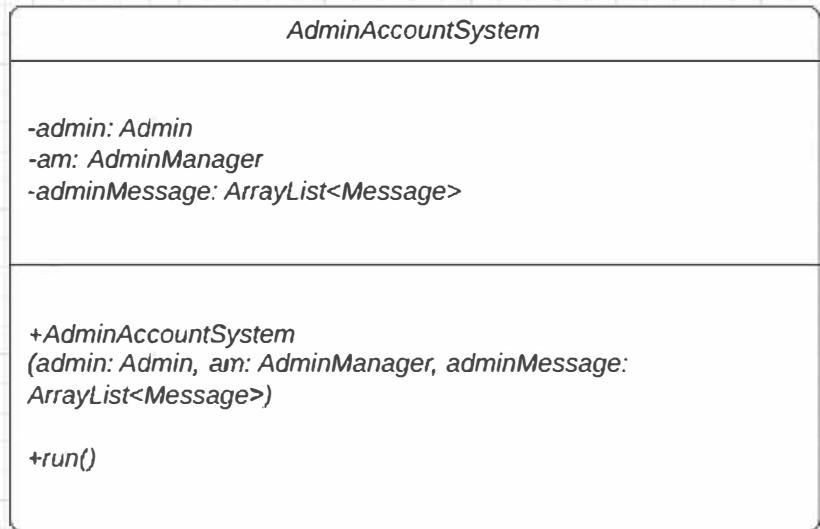
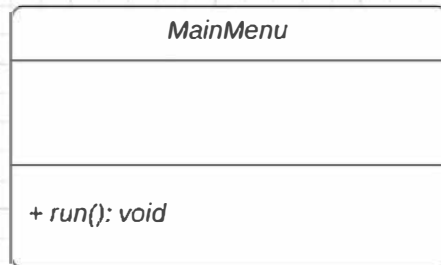
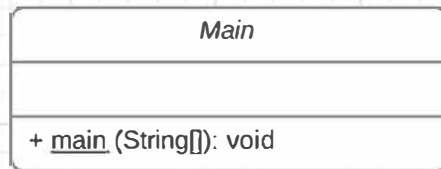
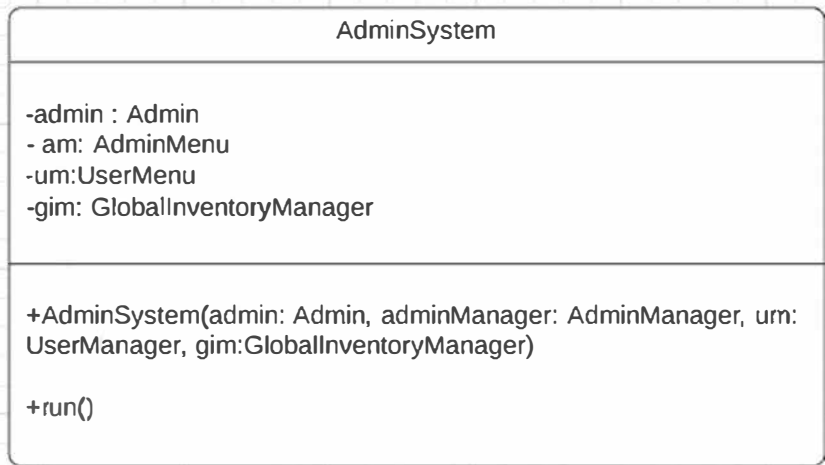
UserPresenter

- + requestFreezeOfUser()
- + userAccountFrozen()
- + promptUserToConfirmTrades()
- + promptUserMenu()
- + createNewItemPrompt(select : int)
- + inputError()
- + setNewPasswordPrompt()
- + printUserTradePartners(tradePartner: String)
- + userInventoryPrompts()
- + userWishlistPrompts()
- + endOfUserInventory()
- + endOfUserWishlist()
- + isEmpty(collection : String)
- + itemRemoved()
- + tooManyIncompleteTrades()
- + tooManyBorrowsVLoans(difference : int)
- + tooManyTradesThisWeek()
- + noTradingPartners()
- + userMenuInfoPrompts()
- + newItemMessageSentToAdmin()
- + tradeRequestSent(username : String)
- + tradeToString(trade : Trade)
- + checkUnconfirmedTradesPrompt()
- + unconfirmedTradeSystemResponse(response : int)
- + itemToString(item : String)
- + noRecentTrades()
- + unfreezeRequestSent()
- + userNotFrozen()
- + emptyPersonalInventoryWhileLoanng()
- + itemNotInOtherUsersWishlist()
- + userTradeHistoryPrompts()
- + userTradeHistoryEndOfIndex()
- + emptyGlobalInventory()
- + loanToOtherUsersPrompt(itemName: String, userName: String)
- + returnToMainMenu()

TradeMenu

- + choosePermTemp()
- + chooseOneOrTwo()
- + itemsAvaliableToTrade(personalInventory: ArrayList<Item>)
- + itemsWantToTrade()
- + enterDate()
- + enterPlace()
- + tradeRequestSent(userB: String)
- + wrongFormat()
- + enteredPastDate()
- + invalidInput()

Controllers



UserMenu

- *currUser: String*
- *adminMessages: ArrayList<Message>*
- *userPresenter: UserPresenter*
- *userManager: UserManager*
- *globalInventoryManager: GlobalInventoryManager*
- *globalWishlistManager: GlobalWishlistManager*
- *tradeManager: TradeManager*

+ *UserMenu(currUser: String, userManager: UserManager, tradeManager: TradeManager, globalInventoryManager: GlobalInventoryManager, globalWishlistManager: GlobalWishlistManager, adminMessages: ArrayList<Message>)*
+ *run()*

AdminBrowsingUsers

- *user: UserManager*
- *browse: AdminBrowsingUsersPresenter*

+ *AdminBrowsingUsers(system: UserManager)*
+ *start(): void*

GlobalInventoryController

+*run(gim: GlobalInventoryManager, UM: UserManager, user: String, TM: TradeManager, GW: GlobalWishlistManager): void*

TradeController

- *input: Scanner*
- *tradeMenu: TradeMenu*
- *date: LocalDateTime*
- *allUsers: UserManager*
- *tradeType: String*
- *done: boolean*
- *tradeRequest: TradeRequest*

+ *TradeController(allUser: UserManager)*
+ *runFromLoan(itemsToTrade: ArrayList<Item>, userA: String, userB: String)*
+ *run(itemsToTradeB: ArrayList<Item>, userB: String)*

UserMessageReplySystem

- *um: UserManager*
- *gi: GlobalInventoryManager*
- *tm: TradeManager*
- *accountUsername: String*
- *rnm: MessageReplyMenu*

+ *UserMessageReplySystem(um: UserManager, gi: GlobalInventoryManager, tm: TradeManager, accountUsername: String)*
+ *run()*

Gateways

GlobalWishlistGateway

+ wishlist: GlobalWishlist

+ GlobalWishlistGateway(filepath: String):
+ writeToFile (filepath: String, wishlistItems2: GlobalWishList)
+ readFromFile (filepath: String, adminMessages: ArrayList
<Message>):
+ getMessages(): ArrayList<Message>

AdminAccountGateways

-filePath: String
-adminMap: HashMap<String, Admin>

+AdminAccountGateways (filePath:String)
+readFromFile()
+saveToFile(adminMap: HashMap<String,Admin>)
+getAdminMap(): HashMap<String, Admin>

AdminMessageGateway

+ messages: ArrayList<Message>

+ AdminAccountMessageGateway(filepath: String):
+ writeToFile (filepath: String, adminMessages: ArrayList
<Message>)
+ readFromFile (filepath: String)
+ getMessages(): ArrayList<Message>

GloballInventoryGateways

+filePath: String
+glManager: GlobalInventoryManager
+gl: GlobalInventory

+GloballInventoryGateways(filePath:String)
+readFromFile()
+saveToFile(gl: GloaballInventory)
+getgl(): GlobalInventory

UserTradesGateway

+ userTrades: HashMap<String, ArrayList <Trade>>

+ UserTradesGateway(filepath: String):
+ writeToFile (filepath: String, userTrades3: HashMap
<String, ArrayList <Trade>>)
+ readFromFile (filepath: String)
+ getUserTrades(): HashMap<String, ArrayList<Trade>>

UserGateway

- mapOfUsers: HashMap<String, User>

+ UserGateway(filepath: String)
+ readFromFile(filepath: String)
+ writeToFile(filepath: String, userObjects: HashMap<String, User>
+ getMapOfUsers()