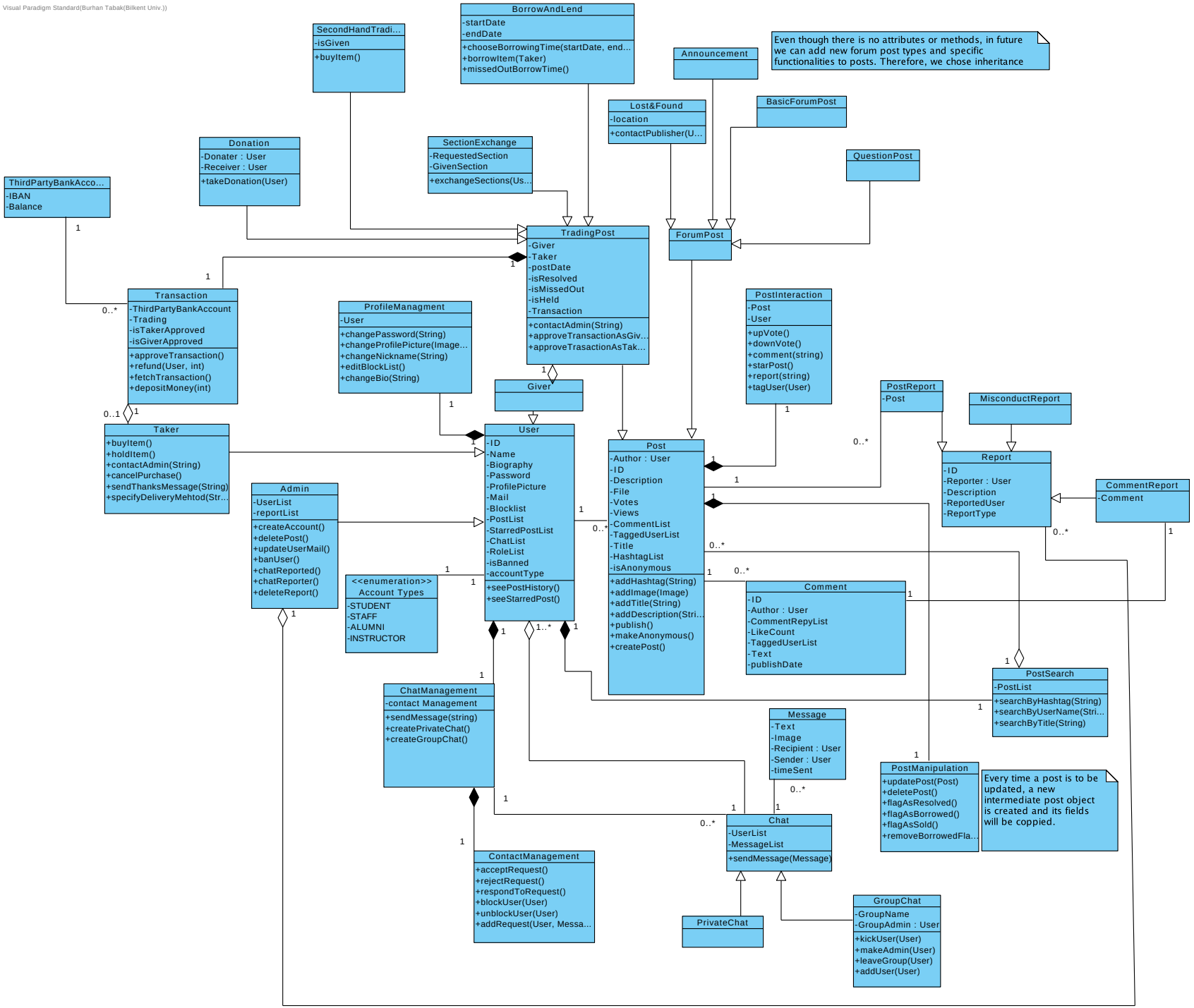
CS319

Deliverable 1
Team 8
Section 1

Tuna Saygın - 22102566
Hüseyin Burhan Tabak - 22102516
Ahmet Tarık Uçur - 22102946
Işıl Özgü - 22102276
Kanan Zeynalov - 22101007



Authentication Package

User: Student, Alumni, Instructor, Staff

1. **Name:** Login
2. **Participating Actor:** User
3. **Entry Condition:** The user has just launched the app, or re-launched the app after not logging out within 5 minutes.
4. **Exit Condition:** The user has successfully logged in.
5. **Flow of Events:**
    5.1. User provides their ID and current password.
    5.2. System checks the user's credentials.
    5.3. User will be checked if he/she is banned.
    5.4. User will be logged in if the credentials are correct.

1. **Name:** UserNoLongerExists
2. **Participating Actor:** Initiated by System
3. **Entry Condition:** The credentials do not match with any current user
4. **Exit Condition:** User is notified about bad credentials
5. **Flow of Events:**
    5.1. System textually notifies user that they cannot log in due to bad credentials

1. **Name:** ForgotPassword
2. **Participating Actor:** User
3. **Entry Condition:**
   User has an account, User forgets password, User remembers their email
4. **Exit Condition:** User is directed to the login page.
5. **Flow of Events:**
    5.1. System sends an email containing a link to change the password.
    5.2. System notifies the user that the email is sent.
    5.3. User approves the password.
    5.4. System redirects the user back.

1. **Name:** UserIsBanned
2. **Participating Actor:**
3. **Entry Condition:** User was banned from the system by an Admin previously
4. **Exit Condition:** User is informed about failed login
5. **Flow of Events:**
    5.1. System textually notifies user that they cannot log in due them being banned

1. **Name:** ChangePassword
2. **Participating Actor:** User
3. **Entry Condition:** User requested a password change.
4. **Exit Condition:** User cancels the change or creates a new password.
5. **Flow of Events:**
    5.1. User enters the new password.
    5.2. System overwrites the password.
    5.3. System sends an email to the user that the password has been changed.

Profile Management Package

1. **Name:** EditAccount
2. **Participating Actor:** User
3. **Entry Condition:** User requests a change in their account.
4. **Exit Condition:** User finishes interacting with Change Password
5. **Flow of Events:**
    5.1. User initiates a password change
    5.2. User is asked their current password.
    5.3. User is asked their new password.
    5.4. User is asked to enter their new password again.

Note: Edit Account only consist of password details. It is different than forgot password because it is when logged in.

1. **Name:** EditProfile
2. **Participating Actor:** User
3. **Entry Condition:** User requests a change in their profile.
4. **Exit Condition:** User finishes interacting with relevant functionality
5. **Flow of Events:**
    5.1. User initiates a bio change or profile picture update
    5.2. System links the user to the profile management page.
    5.3. Profile is updated.

1. **Name:** ChangeBio
2. **Participating Actor:** User
3. **Entry Condition:** User modifies the text written in the bio
4. **Exit Condition:** User saves the changes or cancels.
5. **Flow of Events:**
    5.1. User modifies the text written in the bio
    5.2. User saves the changes
    5.3. System saves the new bio

1. **Name:** UploadProfilePicture
2. **Participating Actor:** User
3. **Entry Condition:** User requests to change their profile picture
4. **Exit Condition:** New photo is saved
5. **Flow of Events:**

5.1. User uploads a photo.
5.2. System checks the file format and accepts the appropriate ones.
5.3. System updates the user profile picture as the new picture

Chat Management Package

1. **Name:** CreateChat
2. **Participating Actor:** User
3. **Entry Condition:** User initiates a chat creation
4. **Exit Condition:** User successfully creates a chat or cancels the initiation
5. **Flow of Events:**
    5.1. User chooses a participant or participants to create a chat
    5.2. System links the user to the relevant use case based on the chat type

1. **Name:** LeaveGroupChat
2. **Participating Actor:** User
3. **Entry Condition:** User must be a participant in the group chat the leave is initiated on
4. **Exit Condition:** User successfully leaves the group chat or cancels the initiation
5. **Flow of Events:**
    5.1. User confirms or cancels the leave initiation
    5.2. System removes that participant from the group chat

1. **Name:** CreateGroupChat
2. **Participating Actor:** Inherited from CreateChat
3. **Entry Condition:** User chooses multiple participants in CreateChat
4. **Exit Condition:** Inherited from CreateChat
5. **Flow of Events:**
    5.1. User selects the participants to be added to the chat
    5.2. System adds the selected participants and the user to a new chat

1. **Name:** CreatePrivateChat
2. **Participating Actor:** Inherited from CreateChat
3. **Entry Condition:**
    3.1. User chooses one participant in CreateChat
    3.2. The selected user is not blocked
4. **Exit Condition:** Inherited from CreateChat
5. **Flow of Events:**
    5.1. User selects the participant to be added to the chat
    5.2. System adds the selected participants and the user to a new chat

1. **Name:** SendMessage
2. **Participating Actor:** User
3. **Entry Condition:** User must be in either a private chat where the chat request was accepted previously, or in a group chat
4. **Exit Condition:** Message is sent
5. **Flow of Events:**
    5.1. User types the message
    5.2. User sends the message
    5.3. System adds the new message to the chat

1. **Name:** IncludeParticipant
2. **Participating Actor:** User
3. **Entry Condition:** User must be in a group chat
4. **Exit Condition:** System adds the indicated participant to the group chat
5. **Flow of Events:**
    5.1. User chooses a new participant to be added to the group chat.
    5.2. User confirms the inclusion of the indicated participant.
    5.3. Group chat invitation is sent to the chosen User.

1. **Name:** SendChatRequest
2. **Participating Actor:** User
3. **Entry Condition:** System created a new private chat
4. **Exit Condition:** System creates a chat request
5. **Flow of Events:**
    5.1. User types new messages to the chat to send alongside the request
    5.2. User confirms the chat request creation
    5.3. System creates the chat request and sends it to the recipient

1. **Name:** UserIsBlocked
2. **Participating Actor:** User
3. **Entry Condition:** Sender is trying to create a private chat with the recipient, and the sender is blocked by the recipient
4. **Exit Condition:** User is informed about the recipient being blocked
5. **Flow of Events:**
    5.1. System informs the user that a private chat cannot be created because the user is blocked by the recipient

Contact Management

1. **Name:** RespondToChat Request
2. **Participating Actor:** User
3. **Entry Condition:** User views the request
4. **Exit Condition:** User replies to the request or leaves it on hold
5. **Flow of Events:**
    5.1. User views the chat request's messages
    5.2. User accepts or declines the request

1. **Name:** AcceptRequest
2. **Participating Actor:** User
3. **Entry Condition:** User views the request
4. **Exit Condition:** User accepts the request
5. **Flow of Events:**
    5.1. User accepts the chat request
    5.2. System adds the user to the chat

1. **Name:** RejectRequest
2. **Participating Actor:** User
3. **Entry Condition:** User views the request
4. **Exit Condition:** User rejects the request
5. **Flow of Events:**
    5.1. User rejects the chat request
    5.2. System deletes the request and the messages

1. **Name:** BlockUser
2. **Participating Actor:** User
3. **Entry Condition:** User requests to block a user.
4. **Exit Condition:** User blocks the indicated user.
5. **Flow of Events:**
    5.1. System adds the indicated user to the blocked users list
    5.2. System prevents the future messages from the indicated user

1. **Name:** UnblockUser
2. **Participating Actor:** User
3. **Entry Condition:** User requests to unblock a user, and the indicated user was blocked
4. **Exit Condition:** User unblocks the indicated user
5. **Flow of Events:**
    5.1. System removes the indicated user from the blocked users list

Admin Panel Package

1. **Name:** CreateAccount
2. **Participating Actor:** Admin
3. **Entry Condition:** Admin opens the admin panel
4. **Exit Condition:** Admin creates an account
5. **Flow of Events:**
    5.1. Admin chooses account type
    5.2. Admin enters an email address
    5.3. Admin enters a user ID
    5.4. Account is created

1. **Name:** CreateStudentAccount
2. **Participating Actor:** inherits from Create Account
3. **Entry Condition:** inherits from Create Account
4. **Exit Condition:** inherits from Create Account
5. **Flow of Events:**
    5.1. Admin chooses student account type
    5.2. Admin enters an email address
    5.3. Admin enters a user ID
    5.4. Account is created

1. **Name:** CreateStaffAccount
2. **Participating Actor:** inherits from Create Account
3. **Entry Condition:** inherits from Create Account
4. **Exit Condition:** inherits from Create Account
5. **Flow of Events:**
    5.1. Admin chooses staff account type
    5.2. Admin enters an email address
    5.3. Admin enters a user ID
    5.4. Account is created

1. **Name:** CreateAlumniAccount
2. **Participating Actor:** inherits from Create Account
3. **Entry Condition:** inherits from Create Account
4. **Exit Condition:** inherits from Create Account
5. **Flow of Events:**
    5.1. Admin chooses alumni account type
    5.2. Admin enters an email address
    5.3. Admin enters a user ID
    5.4. Account is created

1. **Name:** CreateInstructorAccount
2. **Participating Actor:** inherits from Create Account
3. **Entry Condition:** inherits from Create Account
4. **Exit Condition:** inherits from Create Account
5. **Flow of Events:**
    5.1. Admin chooses instructor account type
    5.2. Admin enters an email address
    5.3. Admin enters a user id
    5.4. Account is created

1. **Name:** UpdateUserEmail
2. **Participating Actor:** Admin
3. **Entry Condition:** Admin opens a user in admin panel
4. **Exit Condition:** User email is updated
5. **Flow of Events:**
    5.1. Admin chooses an account by mail or id
    5.2. Admin enters a new email address
    5.3. User email is updated

1. **Name:** ChargeUserForMisconduct
2. **Participating Actor:** Admin
3. **Entry Condition:** Admin receives a misconduct report
4. **Exit Condition:** Account is charged with the specified time
5. **Flow of Events:**
    5.1. Admin receives the misconduct report
    5.2. Admin decides to charge the user
    5.3. Admin chooses the amount of time to charge the user for
    5.4. Admin charges the user

1. **Name:** DeleteAnyPost
2. **Participating Actor:** Admin
3. **Entry Condition:** Admin chooses a post
4. **Exit Condition:** Admin deletes the post of the user
5. **Flow of Events:**
    5.1. Admin chooses a post
    5.2. Admin removes a post due to specific reasons
    5.3. System removes the post from view.
    5.4. Post is deleted.

1. **Name:** BanUser
2. **Participating Actor:** Admin
3. **Entry Condition:** Admin chooses a user
4. **Exit Condition:** Admin successfully bans the user from entering the system
5. **Flow of Events:**
    5.1. Admin chooses a user
    5.2. Admin bans the user
    5.3. System marks the user as banned.
    5.4. User is banned.

1. **Name:** AccountAlreadyExists
2. **Participating Actor:** Admin
3. **Entry Condition:** Admin is trying to create a new user account.
4. **Exit Condition:** System detects an account with same credentials
5. **Flow of Events:**
    5.1. Admin is trying to create a new user account.
    5.2. System finds an account with the same name.
    5.3. Admin is notified.

Post Creation

1. **Name:** CreatePost
2. **Participating Actor:** User
3. **Entry Condition:** User must have the content of the post that he/she wants to create.
4. **Exit Condition:** User created and shared the post
5. **Flow of Events:**
    5.1. User selects which post type to create among (Forum, Lost&Found, trading post)
    5.2. User adds the hashtags.
    5.3. User adds image/images (optional User may add no Images).
    5.4. User tags the wanted user. (optional User may tag no user)
    5.5. User adds a description.
    5.6. Submits the post creation.

1. **Name:** CreateForumPost
2. **Participating Actor:** inherits from Create Post
3. **Entry Condition:** User selects Forum Type
4. **Exit Condition:** inherits from Create Post
5. **Flow of Events:**
    5.1. User selects Forum type.
    5.2. User adds the hashtags.
    5.3. User adds image/images. (optional User may add no Images)
    5.4. User tags the other Users. (optional User may tag no user)
    5.5. User adds description/text.
    5.6. User created the post

1. **Name:** CreateLost&FoundPost
2. **Participating Actor:** inherits from Create Post
3. **Entry Condition:** User selects Lost&Found Post type
4. **Exit Condition:** inherits from Create Post
5. **Flow of Events:**
    5.1. User selects Lost&Found Post type.
    5.2. Lost & Found Label automatically added.
    5.3. Image(s) of the Lost/Found item is added. (optional User may addd no Images)
    5.4. User adds a description.
    5.5. User created the post

1. **Name:** CreateTradingPost
2. **Participating Actor:** inherits from Create Post
3. **Entry Condition:** User selects trading creation
4. **Exit Condition:** inherits from Create Post
5. **Flow of Events:**
    5.1. User selects a trading post creation.
    5.2. User adds hashtag.
    5.3. User gives his IBAN.
    5.4. User adds an image(s) of the item that he/she wants to trade. (optional User may add no Images)
    5.5. User tags other users.(optional User may tag no user)
    5.6. User adds a description.
    5.7. User created the post.

1. **Name:** AttachFile
2. **Participating Actor:** User
3. **Entry Condition:** User adds image while creating post
4. **Exit Condition:** Image is added
5. **Flow of Events:**
    5.1. User selects the image he/she wants to add.
    5.2. Selected Images are uploaded.
    5.3. Image is added.

1. **Name:** TagUser
2. **Participating Actor:**
3. **Entry Condition:** User selects other users to tag while creating post
4. **Exit Condition:** Other user is tagged
5. **Flow of Events:**
    5.1. User is selects another user he/she wants to tag among unblocked users.
    5.2. User tags the user to post.
    5.3. Selected user is tagged.

Post Searching

1. **Name:** SearchPost
2. **Participating Actor:** User
3. **Entry Condition:** User opens the search bar
4. **Exit Condition:** User enters the conditions
5. **Flow of Events:**
    5.1. User opens the search bar.
    5.2. User selects the type of search among SearchByHashtag, SearchByUsername, and SearchByTitle.
    5.3. User searches the post with keywords.
    5.4. User receives the listed results.

1. **Name:** SearchByHashtag
2. **Participating Actor:** Inherits from Search Post
3. **Entry Condition:** Inherits from Search Post
4. **Exit Condition:** Inherits from Search Post
5. **Flow of Events:**
    5.1. User opens the search bar
    5.2. User selects search by hashtag in criteria.
    5.3. User searches the post by their hashtag
    5.4. User receives the listed results

1. **Name:** SearchByTitle
2. **Participating Actor:** Inherits from Search Post
3. **Entry Condition:** Inherits from Search Post
4. **Exit Condition:** Inherits from Search Post
5. **Flow of Events:**
    5.1. User opens the search bar
    5.2. User selects search by title in criteria.
    5.3. User searches the post by their hashtag
    5.4. User receives the listed results

1. **Name:** SearchByUserName
2. **Participating Actor:** Inherits from Search Post
3. **Entry Condition:** Inherits from Search Post
4. **Exit Condition:** Inherits from Search Post
5. **Flow of Events:**
    5.1. User opens the search bar
    5.2. User selects search by username in criteria.
    5.3. User searches the post with keywords
    5.4. User receives the listed results

Comment Interaction

1. **Name:** InteractWithComment
2. **Participating Actor:** User
3. **Entry Condition:** User chooses a comment
4. **Exit Condition:** User leaves the comment
5. **Flow of Events:**
    5.1. User chooses a comment
    5.2. User engages with the comment
    5.3. User leaves the comment

1. **Name:** ReplyComment
2. **Participating Actor:** User
3. **Entry Condition:** User chooses a comment
4. **Exit Condition:** User leaves the comment
5. **Flow of Events:**
    5.1. User chooses a comment
    5.2. User replies to the comment with a message
    5.3. User leaves the comment

1. **Name:** LikeComment
2. **Participating Actor:** User
3. **Entry Condition:** User chooses a comment
4. **Exit Condition:** User leaves the comment
5. **Flow of Events:**
    5.1. User chooses a comment
    5.2. User likes the comment
    5.3. User leaves the comment

1. **Name:** Report
2. **Participating Actor:** User
3. **Entry Condition:** User chooses a comment
4. **Exit Condition:** User leaves the comment
5. **Flow of Events:**
    5.1. User chooses a comment
    5.2. User reports the comment
    5.3. User enters an explanation text about the report
    5.4. User leaves the comment

Post Archiving

1. **Name:** SeePostHistory
2. **Participating Actor:** User
3. **Entry Condition:** User enters the post history tab
4. **Exit Condition:** User closes the tab
5. **Flow of Events:**
    5.1. User enters the post history tab
    5.2. User sees the past posts of themselves as a list

1. **Name:** SeeStarredPost
2. **Participating Actor:** User
3. **Entry Condition:** User enters the starred post tab
4. **Exit Condition:** User closes the tab
5. **Flow of Events:**
    5.1. User enters the starred post tab
    5.2. User sees the starred post list

1. **Name:** PostIsAlreadyDeleted
2. **Participating Actor:** extends from See Post History and See Starred Post
3. **Entry Condition:** extends from See Post History and See Starred Post
4. **Exit Condition:** User is textually warn that the post has been deleted
5. **Flow of Events:**
    5.1. User enters the starred post tab
    5.2. The deleted post is not shown to the user

Forum Post Management Package

1. **Name:** CreateForumPost
2. **Participating Actor:** User
3. **Entry Condition:** User creates a post in forum tab
4. **Exit Condition:** System publishes post
5. **Flow of Events:**
    5.1. User creates a post in forum tab
    5.2. User fills in title, description
    5.3. User adds hashtags
    5.4. User requests to publish the post
    5.5. System publishes the post

1. **Name:** CreateBasicPost
2. **Participating Actor:** Inherits from CreateForumPost
3. **Entry Condition:** Inherits from CreateForumPost
4. **Exit Condition:** Inherits from CreateForumPost
5. **Flow of Events:**
    5.1. User creates a post in basic forum tab (sub tab of Forum tab)
    5.2. User fills in title, description
    5.3. User adds hashtags
    5.4. User requests to publish the post
    5.5. System publishes the post

1. **Name:** CreateQuestionPost
2. **Participating Actor:** Inherits from CreateForumPost
3. **Entry Condition:** Inherits from CreateForumPost
4. **Exit Condition:** Inherits from CreateForumPost
5. **Flow of Events:**
    5.1. User creates a post in the question tab (sub tab of Forum tab)
    5.2. User fills in title, description
    5.3. User adds hashtags
    5.4. User requests to publish the post
    5.5. System publishes the post

1. **Name:** AddAnnouncementPost
2. **Participating Actor:** Inherits from CreateForumPost
3. **Entry Condition:** Inherits from Add CreateForumPost
4. **Exit Condition:** Inherits from Add CreateForumPost
5. **Flow of Events:**
    5.1. User creates a post in announcement tab (sub tab of Forum tab)
    5.2. User fills in title, description
    5.3. User adds hashtags
    5.4. User requests to publish the post
    5.5. System publishes the post
    5.6.

Non-Forum Label Management Package

1. **Name:** CreateTradingPost
2. **Participating Actor:** User
3. **Entry Condition:** User creates a post in trading tab
4. **Exit Condition:** System publishes the post
5. **Flow of Events:**
    5.1. User creates a post in trading tab
    5.2. User adds title, description
    5.3. User adds hashtags
    5.4. User adds price and their IBAN
    5.5. User requests to publish post
    5.6. System publishes the post

1. **Name:** CreateLost&FoundPost
2. **Participating Actor:** User
3. **Entry Condition:** User creates a post in lost&found tab
4. **Exit Condition:** User creates the post
5. **Flow of Events:**
    5.1. User creates a post in lost&found tab
    5.2. User adds title, description
    5.3. User adds hashtags
    5.4. User indicates where the item was found
    5.5. User requests to publish post
    5.6. System publishes the post

1. **Name:** CreateExchangeSectionsPost
2. **Participating Actor:** Inherits from CreateTradingPost
3. **Entry Condition:** Inherits from CreateTradingPost
4. **Exit Condition:** Inherits from CreateTradingPost
5. **Flow of Events:**
    5.1. User creates a post in exchange sections tab (sub tab of Trading tab)
    5.2. User adds title, description
    5.3. User adds hashtags
    5.4. User indicates course and section
    5.5. User adds price and their IBAN
    5.6. User requests to publish post
    5.7. System publishes the post

1. **Name:** CreateDonationPost
2. **Participating Actor:** Inherits from CreateTradingPost
3. **Entry Condition:** Inherits from CreateTradingPost
4. **Exit Condition:** Inherits from CreateTradingPost
5. **Flow of Events:**
    5.1. User creates a post in donation tab (sub tab of Trading tab)
    5.2. User adds title, description
    5.3. User adds hashtags
    5.4. User requests to publish post
    5.5. System publishes the post

1. **Name:** Create2ndHandPost
2. **Participating Actor:** Inherits from CreateTradingPost
3. **Entry Condition:** Inherits from CreateTradingPost
4. **Exit Condition:** Inherits from CreateTradingPost
5. **Flow of Events:**
    5.1. User creates a post in 2nd hand tab (sub tab of Trading tab)
    5.2. User adds title, description
    5.3. User adds hashtags
    5.4. User adds price and their IBAN
    5.5. User requests to publish post
    5.6. System publishes the post

1. **Name:** CreateBorrow&LendPost
2. **Participating Actor:** inherits from CreateTradingPost
3. **Entry Condition:** Inherits from CreateTradingPost
4. **Exit Condition:** Inherits from CreateTradingPost
5. **Flow of Events:**
    5.1. User creates a post in borrow & lend tab (sub tab of Trading tab)
    5.2. User adds title, description
    5.3. User adds hashtags
    5.4. User adds available lending period
    5.5. User requests to publish post
    5.6. System publishes the post

Transaction Management

1. **Name:** ApproveTransaction
2. **Participating Actor:** User, 3rd Party Bank Account
3. **Entry Condition:** Item delivered to User.
4. **Exit Condition:** Trading finished, money is transferred into User
5. **Flow of Events:**
    5.1. Items are delivered to User.
    5.2. Seller approved his/her delivery.
    5.3. User approves the transaction.
    5.4. Money is transferred to the seller from a 3rd Party Bank Account.

1. **Name:** ApproveTransactionAsTaker
2. **Participating Actor:** Taker, 3rd Party Bank Account
3. **Entry Condition:** Item delivered to User.
4. **Exit Condition:**
    4.1. Purchase is approved, money is transferred to the seller by a 3rd Party Bank Account.
    4.2. Purchase is not approved. Transaction information sent to admin. Money is refunded by 3rd Party Bank Account
5. **Flow of Events:**
    5.1. Items are delivered to User.
    5.2. Seller approved his/her delivery.
    5.3. User approves the transaction.
    5.4. Money is transferred to the seller from a 3rd Party Bank Account.

1. **Name:** ApproveTransactionAsGiver
2. **Participating Actor:** Giver
3. **Entry Condition:** User gives the card transactions.
4. **Exit Condition:**
   4.1. Purchase is approved.
   4.2. Purchase is not approved. Transaction by 3rd Party Bank Account.
5. **Flow of Events:**
   5.1. User gets notified that one of his items is sold.
   5.2. User chats with the buyer user for more information.
   5.3. User delivers the item to the buyer via specified methods.
   5.4. User approves that he/she delivered the item.

1. **Name:** FetchTransaction
2. **Participating Actor:** Initiated by System, communicates with 3rd Party Bank Account
3. **Entry Condition:** User gives the card transactions.
4. **Exit Condition:**
   4.1. Purchase is approved.
   4.2. Purchase is not approved. Transaction refunded.
5. **Flow of Events:**
   5.1. User buys the item.
   5.2. Money is transferred to a 3rd Party Bank Account.
   5.3. When the transaction is approved by the buyer, money is transferred to the seller.

1. **Name:** Refund
2. **Participating Actor:** 3rd Party Bank Account
3. **Entry Condition:** Transaction is not approved.
4. **Exit Condition:** Transaction is refunded by 3rd Party Bank Account.
5. **Flow of Events:**
   5.1. User buys the item.
   5.2. Money is transferred to a third party account.
   5.3. When the transaction is approved by the buyer, money is transferred to the seller by a 3rd Party Bank Account.
6. **Special/Quality Requirements:** 3rd Party Bank Account needs to refund money to the user in 3 days at max.

1. **Name:** Refunded
2. **Participating Actor:**
3. **Entry Condition:** Transaction is not approved.
4. **Exit Condition:** Money is refunded to the taker.
5. **Flow of Events:**
    5.1. User buys the item.
    5.2. Money is transferred to a third party account.
    5.3. When the transaction is approved by the buyer, money is transferred to the seller by a 3rd Party Bank Account.
6. **Special/Quality Requirements:** 3rd Party Bank Account needs to refund money to the user in 3 days at max.

1. **Name:** HoldItem
2. **Participating Actor:**
3. **Entry Condition:** User initiated buy action
4. **Exit Condition:**
    4.1. User buys the item
    4.2. User cancels buying
5. **Flow of Events:**
    5.1. User enters the buy use case.
    5.2. Items become in the state of not being bought by other users.

1. **Name:** MissedOut
2. **Participating Actor:**
3. **Entry Condition:** Transaction is held by another user.
4. **Exit Condition:** Buying attempt failed.
5. **Flow of Events:**
    5.1. User attempts to buy an item.
    5.2. Other users are already in the buying state.

Post Interaction

1. **Name:** InteractWithPost
2. **Participating Actor:** User
3. **Entry Condition:** User sees a post.
4. **Exit Condition:** User changed post statistics.
5. **Flow of Events:**
    5.1. User sees a post.
    5.2. Users do actions among Upvote/Downvote, Comment to, or Star the post.
    5.3. Users interact with posts and some of the post information is changed.

1. **Name:** Comment
2. **Participating Actor:** User
3. **Entry Condition:** User attempts to interact with post
4. **Exit Condition:** User made a comment to a post.
5. **Flow of Events:**
    5.1. User attempts to interact with the post.
    5.2. User writes a comment.
    5.3. User commented on a post.

1. **Name:** Up/Downvote
2. **Participating Actor:** User
3. **Entry Condition:** User attempts to interact with post
4. **Exit Condition:** User made a comment to a post.
5. **Flow of Events:**
    5.1. User attempts to interact with the post.
    5.2. User writes a comment.
    5.3. User commented on a post.

1. **Name:** StarPost
2. **Participating Actor:** User
3. **Entry Condition:** User attempts to interact with post
4. **Exit Condition:** User starred a post.
5. **Flow of Events:**
    5.1. User attempts to interact with the post.
    5.2. User starts the post.

1. **Name:** PostCreatorHasAlreadyBlockedYou
2. **Participating Actor:** User
3. **Entry Condition:** User attempts to interact with post
4. **Exit Condition:** User cannot interact with post.
5. **Flow of Events:**
    5.1. User attempts to interact with the post.
    5.2. Post creator has already blocked you.
    5.3. User can not interact with the post.

Post Manipulation Package

1. **Name:** DeletePost
2. **Participating Actor:** User
3. **Entry Condition:** User is viewing a post created by them
4. **Exit Condition:** User deletes the post
5. **Flow of Events:**
    5.1. User requests to delete the post
    5.2. System marks the post as deleted and invisible to others

1. **Name:** NotFound
2. **Participating Actor:** User
3. **Entry Condition:** The user does initiate search operation
4. **Exit Condition:** The system returns search results
5. **Flow of Events:** User could not find searched post

1. **Name:** UpdatePost
2. **Participating Actor:** User
3. **Entry Condition:** User requests a change in their post.
4. **Exit Condition:** User finishes interacting with relevant functionality
5. **Flow of Events:**
    5.1. User makes the preferred changes to the post.
    5.2. User confirms the changes
    5.3. System updates the changed post

1. **Name:** FlagAsResolved
2. **Participating Actor:** User
3. **Entry Condition:** Post is resolved
4. **Exit Condition:** System marks the post as resolved
5. **Flow of Events:**
    5.1. User flags the post as “borrowed” or “sold”
    5.2. System adds the mark
    5.3. System updates the post as resolved

1. **Name:** RemoveBorrowedFlag
2. **Participating Actor:** User
3. **Entry Condition:** User requests to remove the “borrowed” flag
4. **Exit Condition:** System removes the “borrowed” flag from the post
5. **Flow of Events:**
    5.1. User confirms to remove the “borrowed” flag from the post
    5.2. System removes the “borrowed” flag and updates the post accordingly

1. **Name:** FlagAsSold
2. **Participating Actor:** User
3. **Entry Condition:** User requests to flag the post as “sold”
4. **Exit Condition:** System flags the post as “sold”
5. **Flow of Events:**
    5.1. User confirms to flag the post as “sold”
    5.2. System flags the post as “sold” and updates the post accordingly

1. **Name:** FlagAsBorrowed
2. **Participating Actor:** User
3. **Entry Condition:** User requests to flag the post as “borrowed”
4. **Exit Condition:** System flags the post as “borrowed”
5. **Flow of Events:**
    5.1. User confirms to flag the post as “borrowed”
    5.2. System flags the post as “borrowed” and updates the post accordingly

Interact With Trade/Borrow Post

1. **Name:** ContactAdmin
2. **Participating Actor:** User
3. **Entry Condition:** User requests to contact with an admin
4. **Exit Condition:** System sends the contact request to admin
5. **Flow of Events:**
    5.1. User types the necessary messages
    5.2. User confirms the contact request
    5.3. System saves the messages and notifies the admin

1. **Name:** CancelBuying
2. **Participating Actor:** User
3. **Entry Condition:** User changes his/her mind.
4. **Exit Condition:** User cancels buying
5. **Flow of Events:**
    5.1. User changes his/her mind.
    5.2. User cancels the purchase.

1. **Name:** ChooseBorrowingPeriod
2. **Participating Actor:** User
3. **Entry Condition:** User chooses to borrow an item
4. **Exit Condition:** Preferences are saved
5. **Flow of Events:**
    5.1. User enters the time interval that they want to borrow the item
    5.2. System saves the preferences

1. **Name:** BuyItem
2. **Participating Actor:** User
3. **Entry Condition:** User requests to buy the item
4. **Exit Condition:** System notifies the seller of the buy
5. **Flow of Events:**
    5.1. User makes the payment
    5.2. User specifies delivery method among on Campus, or Cargo Delivery.
    5.3. User confirms the purchase
    5.4. System saves the payment
    5.5. System notifies the seller

1. **Name:** TakeDonation
2. **Participating Actor:** User
3. **Entry Condition:** Uses chooses to take donation
4. **Exit Condition:** The donaton is successfully completed, and the item is given to the recipient.
5. **Flow of Events:**
    5.1. User chooses to take donation.
    5.2. User writes a thank message to donator.
    5.3. Donation is completed.

1. **Name:** BorrowItem
2. **Participating Actor:** User
3. **Entry Condition:** User requests to borrow the item on a lend post
4. **Exit Condition:** System notifies the lender of the borrowing request
5. **Flow of Events:**
    5.1. User chooses a borrowing period
    5.2. User confirms the entered information
    5.3. System saves the preferences
    5.4. System notifies the lender of the borrowing request