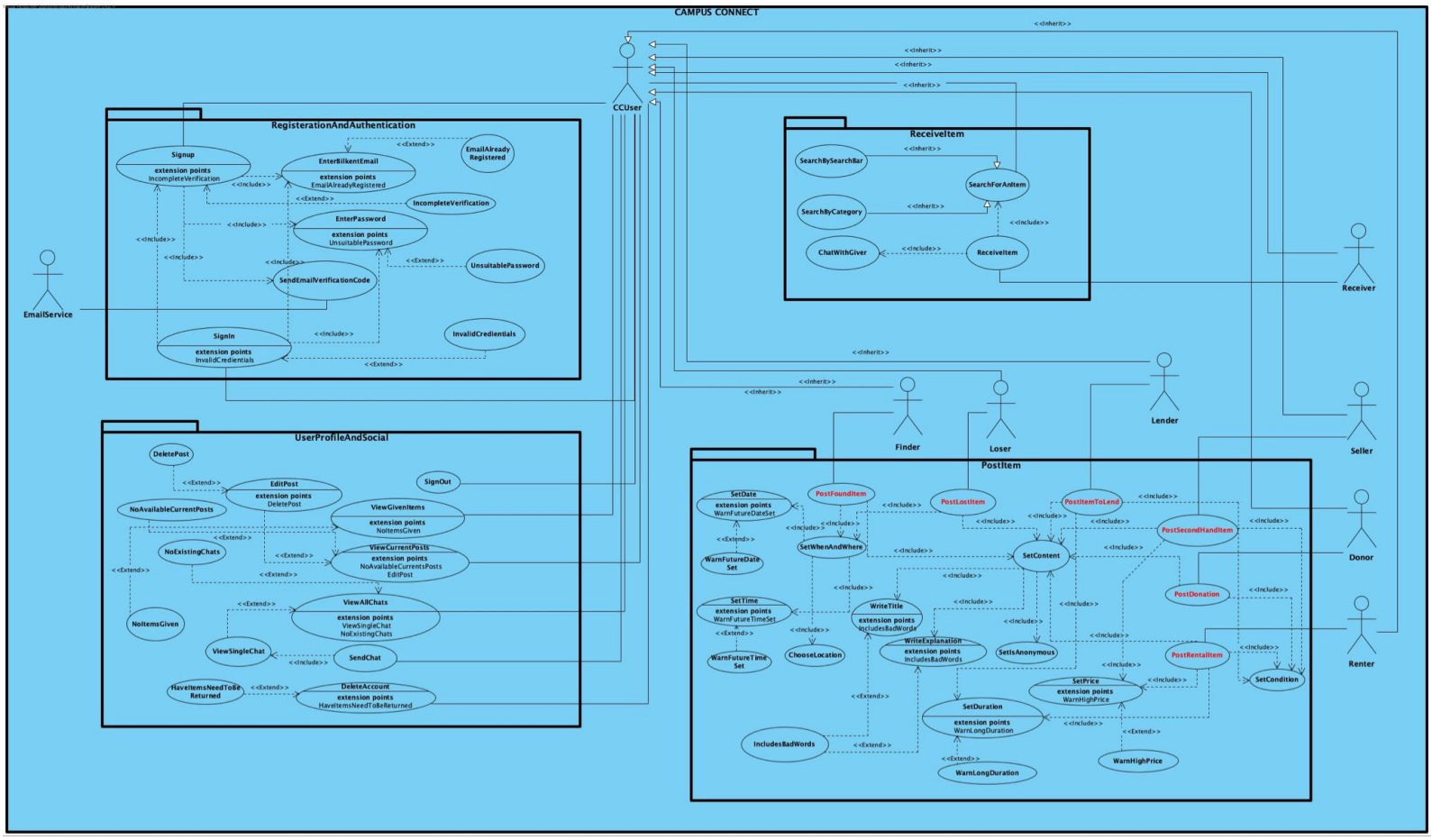
D1-D2 Final Report (12 November 2023)

1. Use Case Diagram of Campus Connect

Use Case Diagram Screenshot



Use Case Diagram Textual Descriptions

RegistrationAndAuthentication Package

Use case name	Signup
Participating actors	CCUser
Flow of events	• The user enters the SignUp case when choosing it on the SignIn case. • The user enters a password (directed to EnterPassword) and Bilkent email (directed to EnterBilkentEmail) to sign up to the application if he/she does not have an account.
Entry condition	• The user chooses the “Sign Up” option instead of signing in. • The user is directed back to this case to complete the registering process after entering password and email.
Exit condition	• Email verification is not completed successfully (directed to IncompleteVerfication) • The user wants to sign up and enter a new Bilkent email (directed to EnterBilkentEmail).

	● The user comes back to this case from EnterBilkentEmail (directed to SendVerificationCode). ● The user wants to sign up and enter a password (directed to EnterPassword). ● The user completes the email verification process and sign up is completed successfully. ● The user quits to sign up.
Quality requirements	The user comes back to this case after entering password and email, and needs to complete email verification to exit this case successfully.

Use case name	SendEmailVerificationCode
Participating actors	EmailService
Flow of events	1. The system sends the user a verification code via the provided Bilkent email. 2. The user enters the verification code. After system checks the code: a. If codes match, verification and signing up process are completed. b. If codes do not match, the user receives another code and the process is repeated after IncompleteVerification warning.
Entry condition	The user has entered the Bilkent email, the system does not detect any accounts with the same email.
Exit condition	● The user completes the verification process. ● The IncompleteVerification warning is opened. ● The user quits to verify their email.
Quality requirements	As long as the user does not enter the correct code, this case is repeated. If the codes match, the sign up process is completed.

Use case name	EnterBilkentEmail
Participating actors	CCUser
Flow of events	1. The user enters a Bilkent email. 2. The system checks if the email already exists. There are four scenarios: a. If the email does not exist, i. the user is directed back to the SignUp case to complete verification when trying to sign up. ii. the user is directed back to the SignIn case and receives a warning(InvalidCredientials). b. If the email exists in the system, i. the user receives the warning case EmailAlreadyRegistered, when trying to sign up. ii. the user is directed back to the SignIn case to complete the sign in process when trying to sign up.
Entry condition	● The user chooses to enter a Bilkent email during the SignUp case. ● The user chooses to enter a Bilkent email during the SignIn case. ● The user closes the EmailAlreadyRegistered warning.
Exit condition	● The user enters his/her email and is directed back to the previous page.

	● The system detects an account with the same email (directed to EmailAlreadyRegistered) when the previous case is SignUp. ● The user is directed to the previous case otherwise.
Quality requirements	The entered Bilkent email needs to be unique in the system, to be directed back to SignUp. If the email is not unique, the warning case EmailAlreadyRegistered is entered. The previous page is recorded and the user is directed back to that page accordingly (explained in Flow of events).

Use case name	SignIn
Participating actors	CCUser
Flow of events	1. The user enters a password (directed to EnterPassword) and Bilkent email (directed to EnterBilkentEmail) to sign in to the application if he/she has an account. 2. The system checks if the user with the given password and email exists. There are two scenarios: a. If the user exists in the system, the sign in process is completed successfully. b. If the user does not exist in the system, the InvalidCredientials warning case is entered.
Entry condition	● The user opens the CampusConnect application. ● The user is directed back to this case to complete the signing in process after entering password and email. ● The user is directed back from InvalidCredientials after closing the alert.
Exit condition	● The user completes signing in after EnterBilkentEmail and EnterPassword. ● System detects that the user account does not exist (directed to InvalidCredientials). ● The user chooses the SignUp option. ● The user quits signing in.
Quality requirements	The user needs to exist in the system for the sign in process to be completed, the InvalidCredientials warning page is opened otherwise. The user is directed back to this case after closing the warning.

Use case name	EnterPassword
Participating actors	CCUser
Flow of events	1. The user enters a password. There are two scenarios: a. Previous case is SignUp: i. If the password is suitable, the user is directed to complete SignUp. ii. If the password is not suitable, an UnsuitablePassword warning opens. The user is asked to re-enter password. b. Previous case is SignIn: i. If the password matches the email(determined by the system database), the sign in process is completed.

	ii. If password does not match the email, the user is directed back to SignIn after InvalidCredientials warning.
Entry condition	● The user wants to enter a password in SignIn. ● The user wants to enter a password in SignUp.
Exit condition	The user is directed to the previous page accordingly (explained in Flow of events).
Quality requirements	The entered password needs to match the email in the system, to be directed back to SignIn. If the password is not suitable, the warning case EmailAlreadyRegistered is entered during SignUp. The previous page is recorded and the user is directed back to that page accordingly (explained in Flow of events).

Use case name	EmailAlreadyRegistered
Participating actors	CCUser
Flow of events	The user receives a warning that he/she needs to enter a unique email.
Entry condition	The system detects that the entered Bilkent email is already enrolled, during the SignUp process.
Exit condition	The user closes the warning.
Quality requirements	The user is directed back to EnterBilkentMail after closing the warning.

Use case name	Incomplete Verification
Participating actors	CCUser
Flow of events	The user is notified when he/she does not enter the correct verification code and is directed back to SendEmailVerificationCode to receive and enter another code.
Entry condition	The email verification can not be completed successfully during SendEmailVerificationCode when the user does not enter the correct code.
Exit condition	The user closes the warning (directed back to SendEmailVerificationCode).
Quality requirements	The user is alerted by this case as long as he/she does not enter the correct verification code in the SendEmailVerificationCode case. The user needs to close the notification in order to continue email verification.

Use case name	Unsuitable Password
Participating actors	CCUser
Flow of events	The user is notified that the entered password does not match the password requirements during the SignUp process in the EnterPassword case.
Entry condition	The user enters an unsuitable password in EnterPassword case when the previous case in EnterPassword is SignUp.
Exit condition	The user closes the alert (directed back to EnterPassword).
Quality requirements	This is an alert case to notify the user about the problem. The user needs to close the notification in order to continue signing up.

Use case name	InvalidCredientials
Participating actors	CCUser
Flow of events	1. The user is notified that the password or Bilkent email they have entered is wrong (during EnterPassword or EnterBilkentEmail accordingly) by an alert message. 2. The user is directed to the SignIn page after closing the alert.
Entry condition	● The system detects that the entered password does not match the account with the given Bilkent email in EnterPassword case when the previous case is SignIn. ● The system detects that the entered email does not exist in the system in EnterBilkentEmail case when the previous case is SignIn.
Exit condition	The user closes the alert message (directed back to the SignIn case).
Quality requirements	This is an alert case to notify the user that they have entered either the password or Bilkent email during the signing in process. The alert is determined by the system database check.

ReceiveItem Package

Use case name	SearchForAnItem
Participating actors	CCUser
Flow of events	The CCUser has two options while searching one is search by category and the other one is search bar with each option user can proceed.
Entry condition	User needs to decide to receive an item and search for it.
Exit condition	User exits from the receive item page and goes to a different option or logs out.
Quality requirements	An option must be selected from two, and proceed with it otherwise this will be a stall.

Use case name	ReceiveItem
Participating actors	Receiver
Flow of events	After searching for an item User must decide to receive it and then the actor becomes a receiver then he/she can proceed with starting a communication with the giver.
Entry condition	Users must select a specific item and decide to get it.
Exit condition	User can change the item he/she wants, can do a different operation in the application or logs out of the application
Quality requirements	Selected items must be in the database and the user must proceed after selecting the database.

Use case name	ChatWithGiver
Participating actors	Receiver, CCUser
Flow of events	After going into this stage it is all dependent on the actors on what to do.
Entry condition	To come to this case the receiver must decide to buy the product.

Exit condition	Receiver can decide to buy the product and end the chat or decide not to do anything at all and logout.
Quality requirements	

Use case name	SearchByCategory
Participating actors	CCUser
Flow of events	At this stage it is just the user who does the search until the decision to receive.
Entry condition	There are two options when searching for an item and this is one of them it is just a matter of decision.
Exit condition	Users may decide to use the search bar or not to search at all.
Quality requirements	The searched, desired item must have the categories set correctly so that it can be found.

Use case name	SearchBySearchBar
Participating actors	CCUser
Flow of events	User will use the search bar to enter some keywords and the application then will proceed with these words on what items to retrieve from the database.
Entry condition	Between the two options about how to search, the user must type something into the search bar for this.
Exit condition	Users may decide to enter a category or not to do any receival at all.
Quality requirements	The keywords that have been entered must exist in the explanation of the items that have been searched.

UserProfileAndSocial Package

Use case name	EditPost
Participating actors	User
Flow of events	**1.** The user selects a post **2.** The user edits any posted content (includes null edit)
Entry condition	User wants to edit a post
Exit condition	The post is edited (includes null edit)
Quality requirements	The changes should be synced quickly across the system

Use case name	DeletePost
Participating actors	User
Flow of events	**1.** The user selects an active post **2.** The user wants to delete the post **3.** The post is deleted from the exchange and the system
Entry condition	User wants to delete a post
Exit condition	The post is purged from the system
Quality requirements	Deletion should require confirmation

Use case name	ViewGivenItems
Participating actors	User
Flow of events	**1.** The user accesses view given items feature **2.** All active given items are displayed
Entry condition	User wants to view their given items
Exit condition	All active given items are displayed
Quality requirements	Loading process should be quick

Use case name	ViewCurrentPosts
Participating actors	User
Flow of events	**1.** The user accesses the view current posts feature **2.** All posted items within all categories are displayed
Entry condition	User wants to view all of their posts in all categories
Exit condition	All post history is displayed
Quality requirements	All inactive posts should be rebalanced in terms of availability

Use case name	ViewAllChats
Participating actors	User
Flow of events	**1.** The user accesses view all chats feature **2.** All chats are displayed
Entry condition	User wants to view their chat history
Exit condition	All history is displayed
Quality requirements	There should be a limit for storing chat histories

Use case name	SendChat
Participating actors	User
Flow of events	**1.** The user sends chat
Entry condition	The user is in a single chat
Exit condition	The chat is sent
Quality requirements	Relaying should be quick

Use case name	DeleteAccount
Participating actors	User
Flow of events	**1.** The user wants to delete their account **2.** All active posts are disbanded **3.** The user account is deleted
Entry condition	User wants to delete their account
Exit condition	The account is deleted with all active posts.
Quality requirements	The deletion process should be secure, and there shouldn't be any ongoing exchange

Use case name	NoAvailableCurrentPosts

Participating actors	User
Flow of events	**1.** The user accesses display current posts feature **2.** The error message is shown
Entry condition	The user accesses the view current posts feature when there is no post
Exit condition	Error prompt is shown
Quality requirements	The error prompt should be displayed quickly

Use case name	NoExistingChats
Participating actors	User
Flow of events	**1.** The user wants to view chats **2.** Error prompt is shown
Entry condition	User wants to view chat history when there is none
Exit condition	Error prompt is shown
Quality requirements	The error prompt should redirect quickly

Use case name	NoItemsGiven
Participating actors	User
Flow of events	**1.** The user wants to display given items **2.** An error message is shown
Entry condition	User displays given items when there is none
Exit condition	Error prompt is displayed
Quality requirements	Error prompt should redirect the user

Use case name	ViewSingleChat
Participating actors	User
Flow of events	**1.** The user selects a specific chat from all chats **2.** The chat is loaded
Entry condition	User selects a chat from all chats
Exit condition	The chat is displayed
Quality requirements	The chats should be accessible and visually appealing

Use case name	HaveItemsNeedToBeReturned
Participating actors	User
Flow of events	**1.** The user wants to delete their account **2.** The required items are listed **3.** Items are returned **4.** User's account is deleted
Entry condition	The user needs to delete their account but has items to be returned
Exit condition	Items are returned, and the user account is deleted
Quality requirements	The system should protect the item creditors

Use case name	SignOut
Participating actors	User

Flow of events	**1.** The user accesses the "View Profile" feature **2.** User logs out
Entry condition	The user is logged in. (This also confirms that the user is a valid user.) The user is on the profile page
Exit condition	Return to sign-in/register page
Quality requirements	The log-out should be quick

PostItem Package

Use case name	PostFoundItem
Participating actors	Initiated by the Finder
Flow of events	1. The finder sets the main content (SetContent) of the found item. Such as its title and explanation. Also chooses to stay anonymous or not. 2. The finder indicates where and when he/she found the item (SetWhenAndWhere) by choosing a location among the provided list of locations, entering the date and exact time he found the item.
Entry condition	● The finder must be signed up and logged in to the system.
Exit condition	● Found item is posted to the Campus Connect System and available for any interactions.
Quality requirements	

Use case name	PostLostItem
Participating actors	Initiated by the Loser
Flow of events	1. The loser sets the main content (SetContent) of the lost item. Such as its title and explanation. Also chooses to stay anonymous or not. 2. The loser indicates where and when he/she lost the item (SetWhenAndWhere) by choosing a location among the provided list of locations, entering the date and exact time he lost the item.
Entry condition	● The loser must be signed up and logged in to the system.
Exit condition	● Lost item is posted to the Campus Connect System and available for any interactions.
Quality requirements	

Use case name	PostItemToLend
Participating actors	Initiated by the Lender
Flow of events	1. The lender sets the main content (SetContent) of the lent item. Such as its title and explanation. Also chooses to stay anonymous or not. 2. The lender indicates the duration (SetDuration) that he/she wants to lend the item for by specifying a duration shorter than year.

	3. The lender indicates the current condition (SetCondition) of the item to be lent from a provided list of these values: Like New, Very Good, Good, Fair, Poor.
Entry condition	● The lender must be signed up and logged in to the system.
Exit condition	● Lended item is posted to the Campus Connect System and available for any interactions.
Quality requirements	

Use case name	PostSecondHandItem
Participating actors	Initiated by the Seller
Flow of events	1. The seller sets the main content (SetContent) of the second hand item. Such as its title and explanation. Also chooses to stay anonymous or not. 2. The seller indicates the price (SetPrice) of the item that is under some price limitation. 3. The seller indicates the current condition (SetCondition) of the second hand item from a provided list of these values: Like New, Very Good, Good, Fair, Poor.
Entry condition	● The seller must be signed up and logged in to the system.
Exit condition	● Second hand item is posted to the Campus Connect System and available for any interactions.
Quality requirements	

Use case name	PostDonation
Participating actors	Initiated by the Donor
Flow of events	1. The donor sets the main content (SetContent) of the donated item. Such as its title and explanation. Also chooses to stay anonymous or not. 2. The donor indicates the current condition (SetCondition) of the item to be donated from a provided list of these values: Like New, Very Good, Good, Fair, Poor.
Entry condition	● The donor must be signed up and logged in to the system.
Exit condition	● Donated item is posted to the Campus Connect System and available for any interactions.
Quality requirements	

Use case name	PostRentalItem
Participating actors	Initiated by the Renter
Flow of events	1. The renter sets the main content (SetContent) of the rental item. Such as its title and explanation. Also chooses to stay anonymous or not.

	2. The renter indicates the price (SetPrice) of the item that is under some price limitation. 3. The renter indicates the current condition (SetCondition) of the second hand item from a provided list of these values: Like New, Very Good, Good, Fair, Poor. 4. The renter indicates the duration (SetDuration) that he/she wants to rent the item for by specifying a duration shorter than year.
Entry condition	● The renter must be signed up and logged in to the system.
Exit condition	● Rental item is posted to the Campus Connect System and available for any interactions.
Quality requirements	

Use case name	SetContent
Participating actors	Initiated by the Lender, Donor, Renter, Loser, Finder, Seller
Flow of events	1. Actor specifies if he wants to stay anonymous or not (SetIsAnonymous). 2. Actor gives a title to the item to be posted (WriteTitle). 3. Actor writes an explanation of the item (WriteExplanation).
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Content is successfully set without any missing arguments. ● No bad words are used while writing the title and the explanation.
Quality requirements	

Use case name	SetCondition
Participating actors	Initiated by the Lender, Donor, Renter, Seller
Flow of events	1. The current condition of the item to be posted is selected by the actor from a provided list of these values: Like New, Very Good, Good, Fair, Poor.
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Conditional status of the item is selected from the provided values.
Quality requirements	

Use case name	SetPrice
Participating actors	Initiated by the Renter, Seller
Flow of events	1. Price of the item is set.
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Price must be under some price limit.

Quality requirements	

Use case name	SetDuration
Participating actors	Initiated by the Renter, Lender
Flow of events	1. Duration to give the item away is selected by the poster actor (either renter or lender).
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Give away duration must be shorter than a year.
Quality requirements	

Use case name	SetWhenAndWhere
Participating actors	Initiated by the Loser, Finder
Flow of events	1. Exact date of losing/finding is specified (SetDate). 2. Exact time of losing/finding is specified (SetTime). 3. The actor selects from the given list the location in which he found the item (SetLocation).
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Date and time must be valid and must have passed.
Quality requirements	

Use case name	SetDate
Participating actors	Initiated by the Loser, Finder
Flow of events	1. Exact date of losing/finding is specified.
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Date must be valid and must have passed.
Quality requirements	

Use case name	SetTime
Participating actors	Initiated by the Loser, Finder
Flow of events	1. Exact time of losing/finding is specified.
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Time must be valid and must have passed.
Quality requirements	

Use case name	ChooseLocation
Participating actors	Initiated by the Loser, Finder
Flow of events	2. Place where the actor lost or found the item is specified.
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Location is chosen from the provided list.
Quality requirements	

Use case name	WriteTitle
Participating actors	Initiated by the Lender, Donor, Renter, Loser, Finder, Seller
Flow of events	1. Actor gives a self-explanatory title for the item he/she is currently posting.
Entry condition	● Actor must be posting an item. ● Actor must be signed up and logged in to the system.
Exit condition	● Title should not contain any bad words.
Quality requirements	

Use case name	WriteExplanation
Participating actors	Initiated by the Lender, Donor, Renter, Loser, Finder, Seller
Flow of events	1. The actor writes an explanatory text for the item he/she is currently posting.
Entry condition	● Actor must be posting an item. ● The actor must be signed up and logged in to the system.
Exit condition	● Explanations should not contain any bad words.
Quality requirements	

Use case name	SetIsAnonim
Participating actors	Initiated by the Lender, Donor, Renter, Loser, Finder, Seller
Flow of events	1. After the actor posts, he will determine whether his name will be visible to outsiders or not by this functionality.
Entry condition	● Actor must be posting an item. ● The actor must be signed up and logged in to the system.
Exit condition	The actor stated whether he wanted to be anonymous or not.
Quality requirements	

Use case name	WarnHighPrice

Participating actors	Communicates with the Renter or Seller
Flow of events	1. An alert is displayed by the system saying the price entered by the participating actor is not in the provided price range.
Entry condition	● This use case extends the SetPrice use case. It is initiated by the system when the Renter or Seller tries to specify a higher price than the determined highest price limit.
Exit condition	● Warning about the high price is displayed to the participating actor. Typically, a renter or seller of the system.
Quality requirements:	The price limit should be optimal. Neither too high nor too low. It should be determined so that users do not misuse the system.

Use case name	Warn Long Duration
Participating actors	Communicates with the Lender or Renter
Flow of events	1. An alert is displayed by the system saying the duration entered by the participating actor is above the provided duration limit. Which is 1 year by default.
Entry condition	● This use case extends the SetDuration use case. It is initiated by the system when the specified duration to lend or rent the item is longer than a year.
Exit condition	● Warning about the long duration is displayed to the participating actor. Typically a renter or lender of the system.
Quality requirements	

Use case name	IncludesBadWords
Participating actors	Communicates with the Lender, Renter, Donor, Seller, Loser or Finder
Flow of events	1. An alert is displayed by the system saying the title or explanation written by the participating actor includes some bad word that must be removed before the item is posted to the system.
Entry condition	● This use case extends the SetTitle and SetExplanation use cases. It is initiated by the system when the title or explanation entered includes bad words.
Exit condition	● Warning about the bad words is displayed to the participating actor.
Quality requirements	

Use case name	WarnFutureTimeSet
Participating actors	Communicated with the Loser o Finder
Flow of events	1. An alert is displayed by the system saying the time entered is not valid because it is representing future time.
Entry condition	● This use case extends the SetTime use case. It is initiated by the system when the time entered represents the future.

Exit condition	● A warning about the time not representing the correct information was displayed on the screen.
Quality requirements	

Use case name	WarnFutureDateSet
Participating actors	Communicated with the Loser or Finder
Flow of events	1. An alert is displayed by the system saying the date entered is not valid because it represents a future date.
Entry condition	● This use case extends the SetDate use case. It is initiated by the system when the date entered represents the future.
Exit condition	● A warning about the date not representing the correct information was displayed on the screen.
Quality requirements	

2. NFR Non-Functional Requirements

User-Friendly Interface

This application provides a detailed, glamorous, and user-friendly interface. Our objective was to merge these 3 makings in the same application. We designed our web application to be as minimal as possible for ease of the user. Our goal is to present the web app's elements in a polished and tidy manner, aiming for a clean and straightforward appearance. Every component should be easily accessible, ensuring a user-friendly experience for a more enjoyable interaction. On this objective, we chose reactive as our main front-end development environment due to its richness in libraries and its variety.

Safety

While designing our application, we added some features that will protect the user's privacy to ensure that using this application does not bring any safety/security problems. One of the most obvious security enhancements that is being applied in our application is e-mail verification in the sign-up stage which is for ensuring a safe trading environment for all of the users that are using this application.

Maintainability

The student sales platform at Bilkent University is designed with maintainability in mind. Utilizing Object-Oriented Programming principles ensures a modular and adaptable structure, simplifying the addition of new features or quick bug fixes. This approach safeguards the main codebase from disruptions during optimisations, ensuring smooth and efficient maintenance, the technical side of the maintainability issue is using AWS and Spring systems which will help our applications maintainability with verified programs that have been in use by millions of developers.

User Satisfaction

Without a user, our application is useless the more users interact with our application, the better it is for us because more users mean more posts and more posts mean more interaction inside the application, so user satisfaction is one of our main non-functional requirements whilst designing this web-application. In this regard, we are trying to design an application by looking at the application designs and functions of e-commerce sites in use today.

3. Tech Stack

We will use React.js for the frontend, Java Spring for the backend, and PostgreSQL for the database in our application:

React.js for Frontend Implementation as a JavaScript Framework: React.js is chosen for its efficiency and flexibility in building interactive user interfaces. It allows for the creation of dynamic and responsive web applications with its component-based architecture.

Java Spring for Backend Development: Java Spring is selected for its robust framework that simplifies the development of enterprise applications. It offers comprehensive infrastructure support for building complex backend applications with ease and efficiency.

PostgreSQL for Database Management: PostgreSQL is used as the database due to its advanced features, reliability, and strong compliance with SQL standards. It is a powerful open-source relational database system, known for its scalability and data integrity.

4. Sequence Diagrams

Scenario 1: Getting an Item

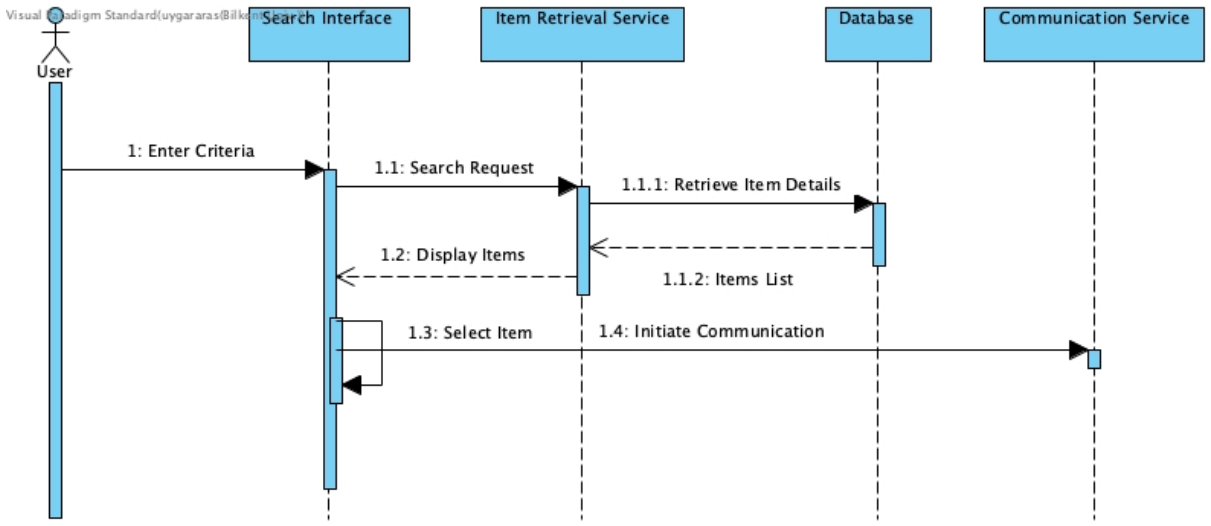


Fig. 1: Sequence Diagram to Demonstrate How Users Get Items

When a user decides the get an Item he/she enters a criteria to the search bar in the search interface and then the search request is sent to the Item Retrieval Service, the service is in communication with the database and from the database it gets the item list to be showed in Search Interface, from here select item can be initiated if user likes it then he/she can initiate a communication.

Scenario 2: Posting an Item

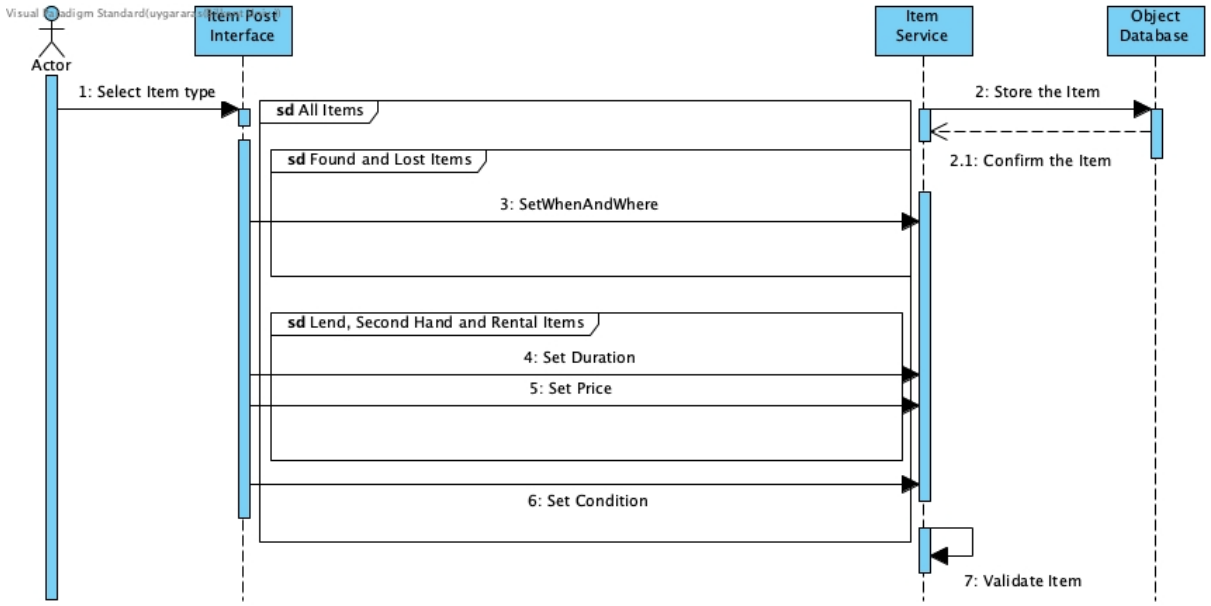


Fig. 2: Sequence Diagram to Demonstrate How Users Post Items

This diagram is a comprehensive one it shows how an item can be listed on the application how an item can be interacted in multiple ways there are 2 sub packages and 1 general one one sub is for Found and Lost items which include SetWhenAndWhere and one for the other type of items that can be posted there is also a general package which has a property of set condition which is general in all item cases.

Scenario 3: Actions That the User Can Perform From His/Her Profile

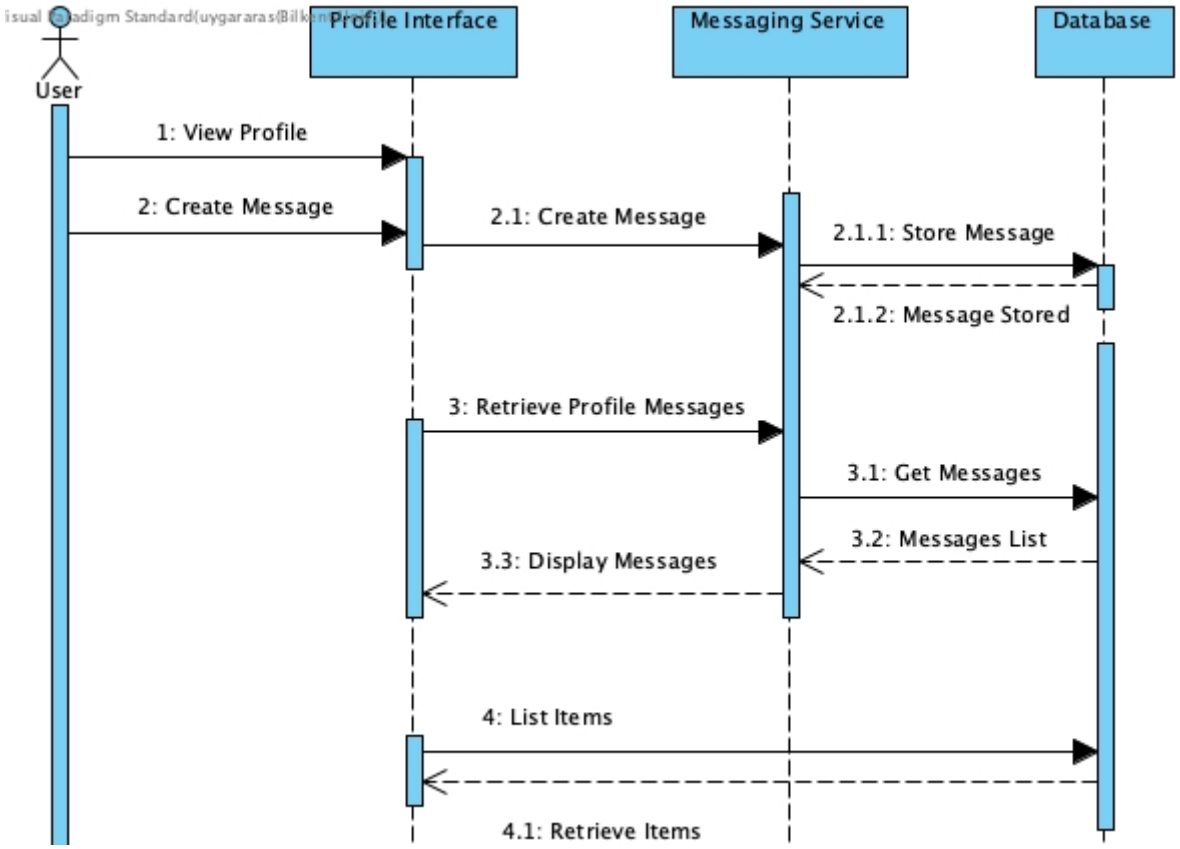


Fig. 3: Sequence Diagram to Demonstrate How Users Send and Search for Messages, Display Sold or Current Items From User Profile Page

This is the sequence diagram of the user profile settings screen. A user can show all of the items that he/she listed from the beginning, also there is a conversations part of the application where every user has a list of previous chats and in the diagram it is shown that the user can interact with these conversations in multiple ways.

Scenario 4: Authentication Flow While Sign In

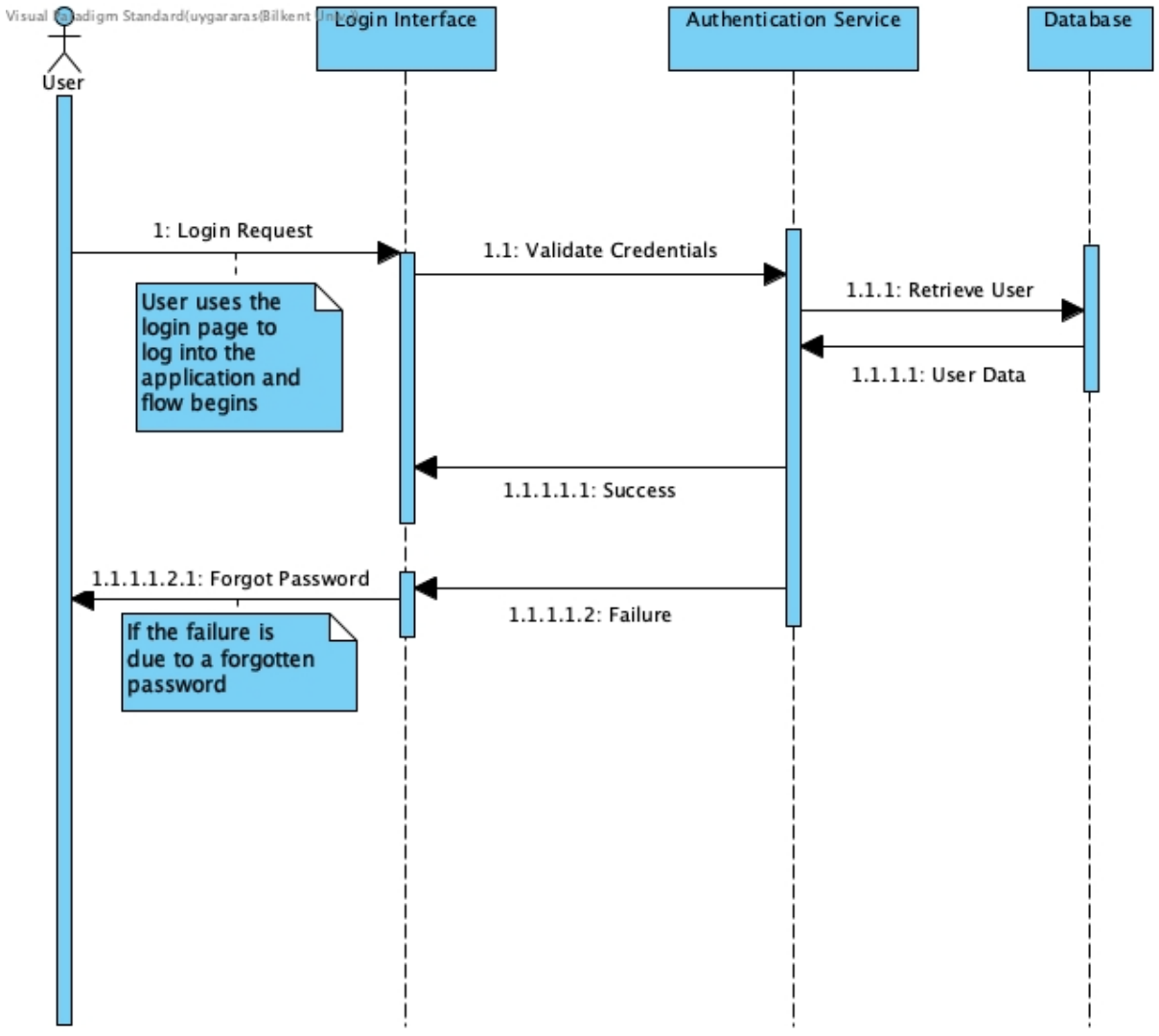


Fig. 4: Sequence Diagram to Demonstrate How Users Authentication is Being Done

This is the sequence diagram for how the authentication flow of the application is being executed at the login stage of the application. There are explanatory messages in the diagram so it is a simple and clear one.

5. Activity Diagrams

Activity Diagram to Demonstrate Authentication Process

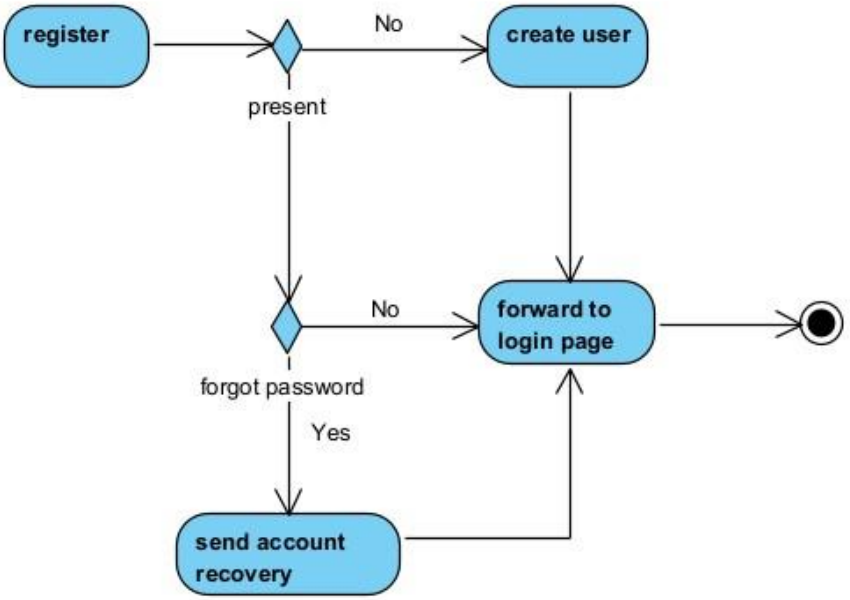


Fig. 4: Activity Diagram That Shows the Registering Process

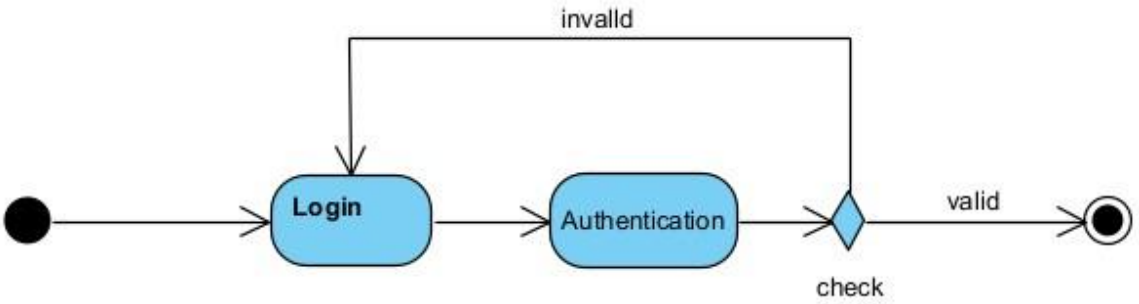


Fig. 5: Activity Diagram That Shows the Login Process

The activity diagram shows how a client enters the CampusConnect.The first decision is whether the user wants to "Sign In." If they choose to sign in, the system requires the user password and the user name. If the correct credentials are provided, the user finishes the authentication process. If not, the process offers them the "Register" if they wish to do so with their chosen password and Bilkent identity number. This way, a new user is created and added to the system. After either signing in or registering the authentication process concludes.

Activity Diagrams to Demonstrate Exchange Process

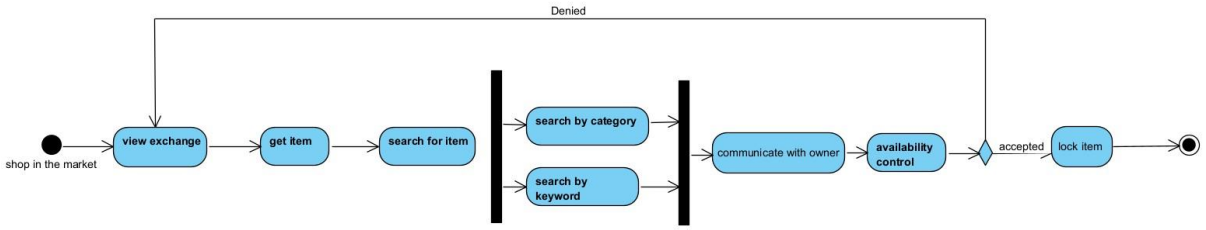


Fig. 6: Activity Diagram that Shows the Shopping Proces

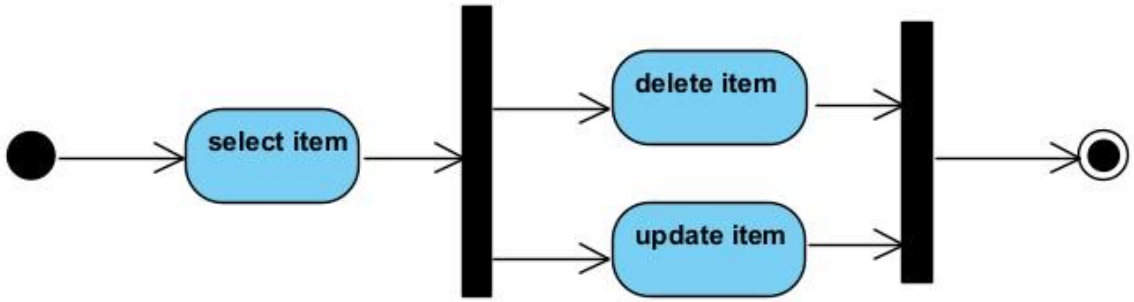


Fig. 7: Activity Diagram that Shows Updating Process

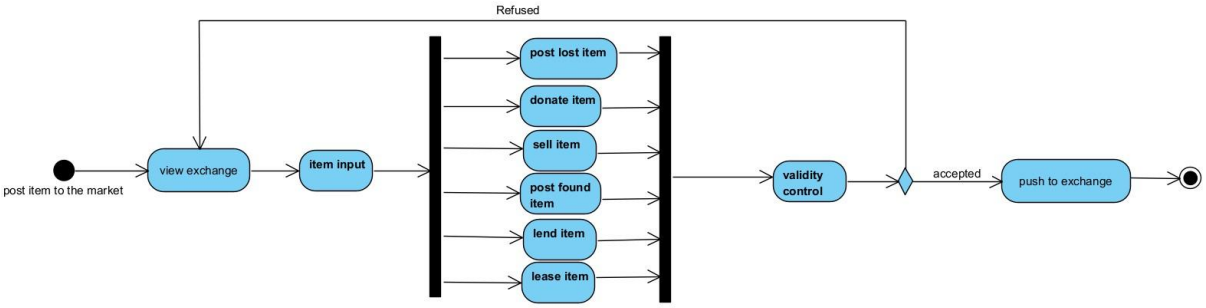


Fig. 8: Activity Diagram that Shows Inputting an item to the Exchange

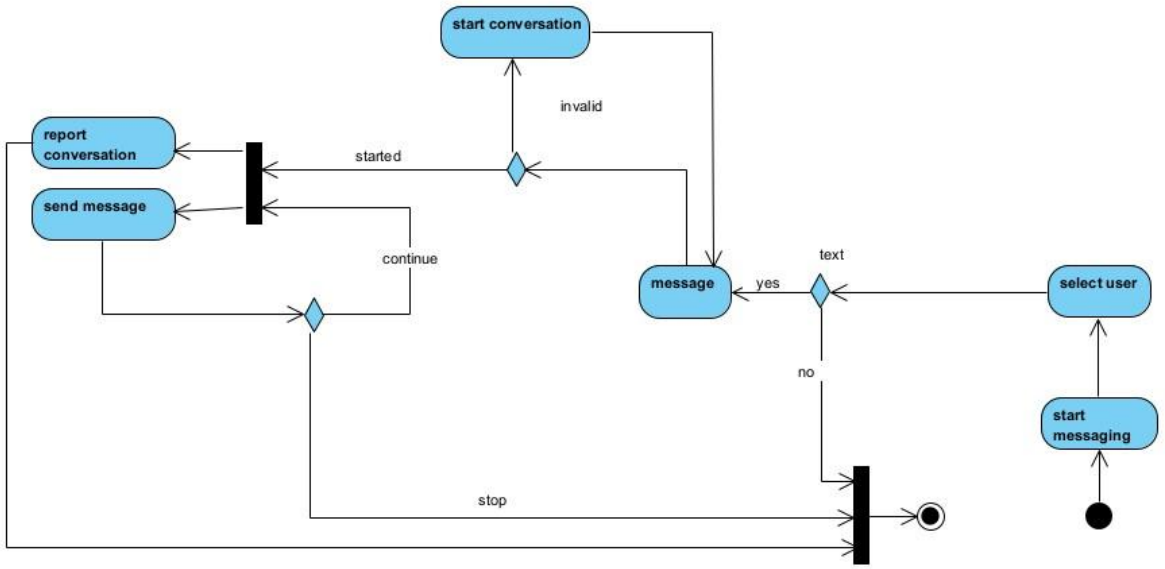


Fig. 9: Activity Diagram that Shows Messaging

In this activity diagram, imagine yourself stepping into the bustling digital marketplace known as Exchange. After entering the exchange, as shown in the activity diagram, one has two options for exploring existing items: searching for items by category or by using specific keywords. Once the user discovers an item that captures their interest, it's time to engage with the seller. This part of the journey is akin to striking up a conversation with a merchant at a traditional marketplace. You inquire about the product, negotiate terms, and ensure it's the right fit for your needs. This is all virtual, thanks to CampusConnect. Now comes a critical juncture in the story: the validation of the item. Both you, as the prospective buyer, and the seller must agree that the item is as described and authentic. It's the digital equivalent of a handshake deal that seals the transaction, locking the item for your purchase.

But the exchange isn't just about taking out items; it's a virtual Bilkent marketplace where you can also contribute. You have a range of options: you can post lost items, donate your belongings, sell items, share found items, lend out items, or even lease items to others. Think of this as setting up your own stall in the marketplace, each representing a unique offering, be it a service or a product. However, your journey as a contributor doesn't end with posting. Before your listed item goes live in the marketplace, it undergoes a validity control process. Once your posted item passes the validity control, it's like watching your stall open for business. Your offering is available for all to see, and the marketplace is richer for your contribution. In this activity diagram, it has been shown how invaluable Bilkent users can explore, interact, trade, and share, all while maintaining trust and transparency in the process. Each user's journey contributes to the ever-evolving story of this online Bilkent-only and Bilkent-specific virtual bazaar.

Activity Diagram to Demonstrate In-Profile Actions

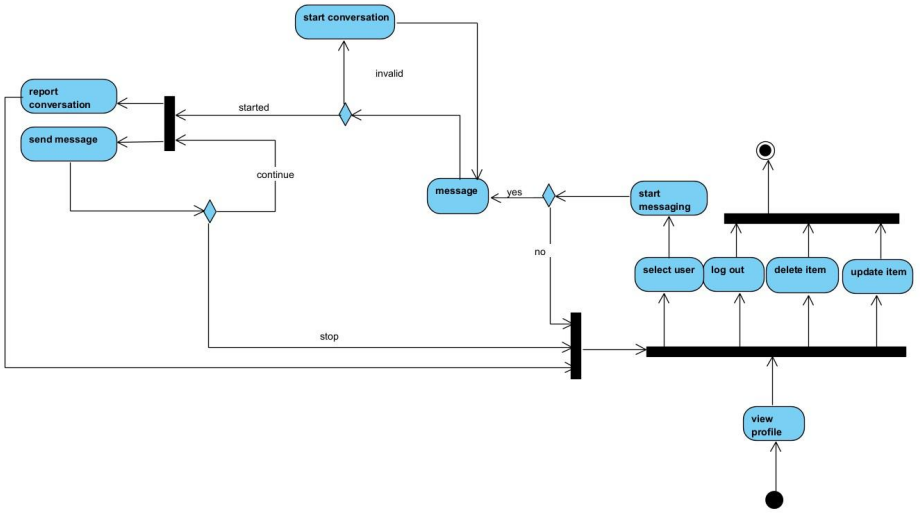


Fig. 6: Activity Diagram That Shows Actions Which Users Can Perform From Their Profiles

In this activity diagram, it has been shown how a user profile page within the CampusConnect performs. This page is not merely a static backdrop; it's a dynamic space filled with messaging features and inventory management tools. As you explore this digital realm, you encounter a multitude of options. In the profile page the user can either manage their inventory or message with another user by selecting and searching them.

The user can start their journey from the profile page, which serves as your central hub for most of the user specific interactions. From there, you face various choices. If the client is ready to conclude their session, the "Sign Out" option acts as their digital exit. Choosing "Test with Other Users" is akin to opening a door to engage with fellow platform users, fostering connections and communication. Alternatively, if the client comes across a conversation that needs attention, "Report Conversation" serves as their lifeline to seek help in addressing issues. With this way, the user can request the banning of the other user if they have malicious behavior.

Furthermore, should the user need to declutter their digital space by removing content from their posted items, "Delete Item" acts as their virtual broom. On the other hand, "Update Item" enables the user to fine-tune and refresh the designated item’s content. Client’s journey within the profile page concludes when they have accomplished the action(s) they have selected. As it has been shown in the activity diagram the profile page, in essence, becomes a dynamic gateway to a range of interactions and tasks, with each choice offering a different facet of CampusConnect’s online experience. Users can safely sign out from the platform via their special and unique profile page securely.

6. Class Diagram

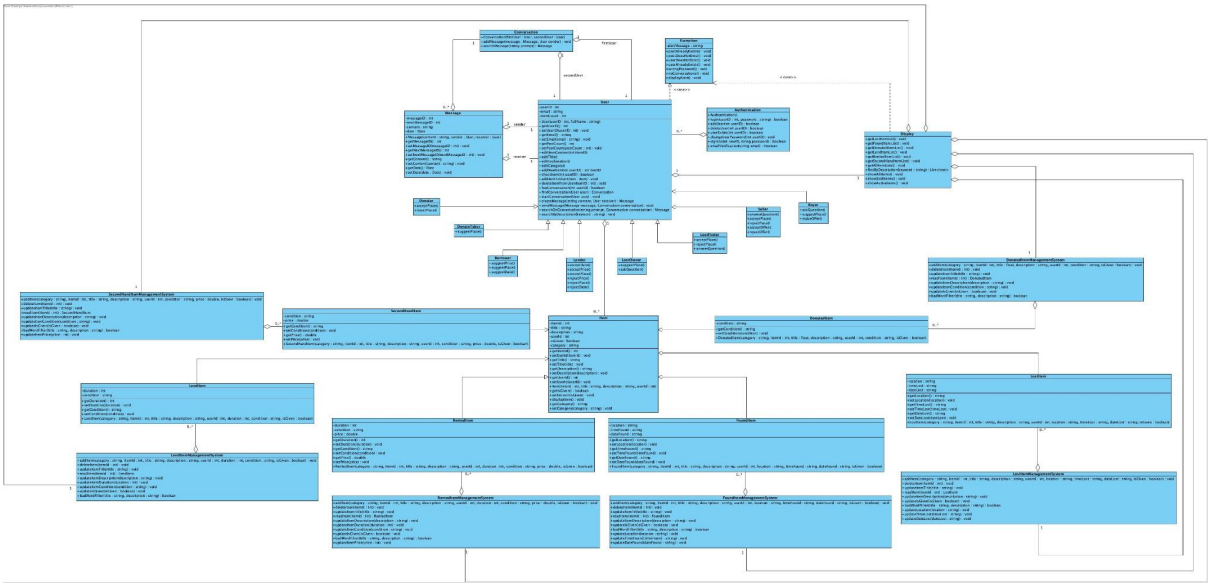


Fig. 7: Class Diagram of Campus Connect

This is the class diagram of Campus Connect that shows the inheritance and other relations of required classes to implement the application. Most processes are to be handled in multiple classes with shown operations and attributes. For example, the Display class uses Item class to show the unsold items in different categories and it is used by the User class to be called when the user wants to view items of a specific category. Classes and their relations with each other are shown by arrows along with their attributes and operations in detail, the multiplicities between classes are also indicated. Different user types to implement unique functionalities for different processes are shown as subclasses of User class. Parameters and return types of operations are also indicated for each class.

7. State Diagrams

State Diagram to Demonstrate Login Process

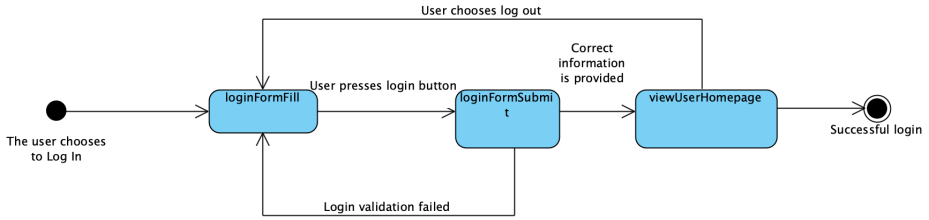


Fig. 8: State Diagram of Login Process

As demonstrated, the state diagram above contains information about the login process. Diagram is dependent on the loginFormFill state which connects to the loginFormSubmit state, establishing the connection between the user interface and database to check if any account is found containing given information in the user table. If there is a match, the user logs in to their account successfully and is directed to the homepage. If the system fails to

find a matching user with the information provided, the user is directed back to the login form state to fill in the information again, until a match is found. Another situation would be the user logging out after accessing their homepage.

State Diagram to Demonstrate Sign Up Process

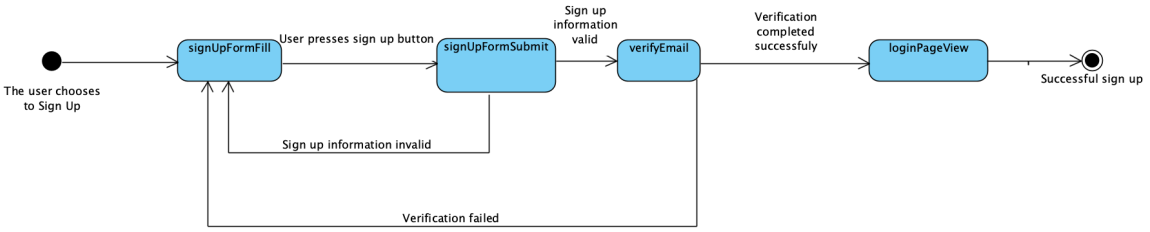


Fig. 9: State Diagram of Sign Up Process

This is the state diagram of the Sign Up process, as the user chooses to sign up after opening the application, they are to fill a sign up form and enter their name, Bilkent ID and password of choice. Then they make a Sign Up Request to the system. If the system does not detect an existing account with the same ID, the user signs up successfully. Otherwise, they are directed to fill the form again with another Bilkent ID or they can quit signing up. The process also contains an email verification state for safety measures.

State Diagram to Demonstrate Process of Posting an Item to the System

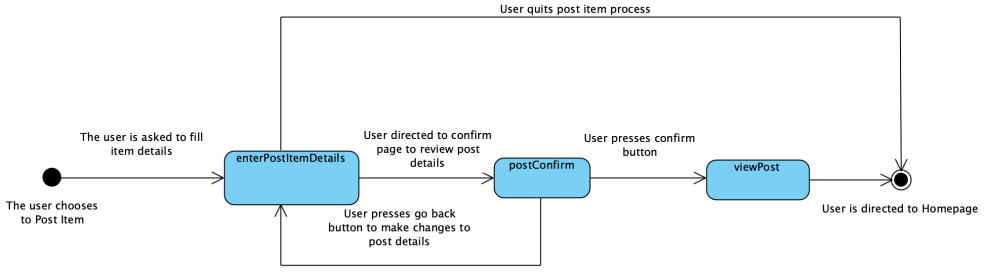


Fig. 10: State Diagram for Posting an Item

This state diagram provides information about the possible states in the situation of posting an ad to the system. The user is directed to the fill post item details state once the process begins, where detailed descriptions and qualifications are entered by the user to filter the desired post item and place it in the correct classification. Next state is established after the form is filled where the user is asked to verify confirmation about posting the item, the two factor system is to prevent users from misclicking the post button and an unfilled form is submitted. Next state is viewing the post that is uploaded to the system after the user confirms the form is filled. An alternative option is available to exit the diagram, where the user quits posting the item and is directed to the homepage.

State Diagram to Demonstrate Process of Searching and Getting an Item from the System

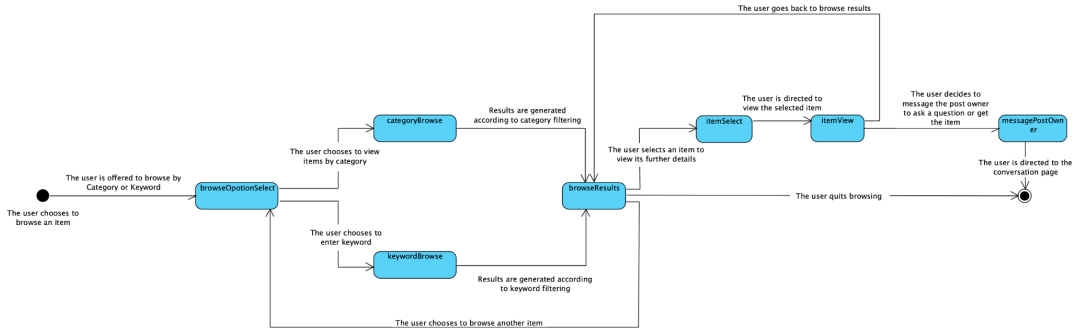


Fig. 11: State Diagram for Searching and Getting an Item

This is the state diagram for browsing and getting an item. When the user clicks to the search buttons on the Homepage, they are to choose whether to search for a keyword or view a specific item category of choice. They are directed to entering the keyword if they choose to do so, or to choosing the category among 6 categories of items. Then the system lists all the searching results for users to view. After viewing the results, the user can quit browsing and go back to the homepage, or select a specific item to view in detail. If they select such an item, they view its photos and other information in further detail. After viewing an item, the user can go back to view their searching results or can message the item post owner to get the item or to ask questions, and quit the state.

8. Mockups

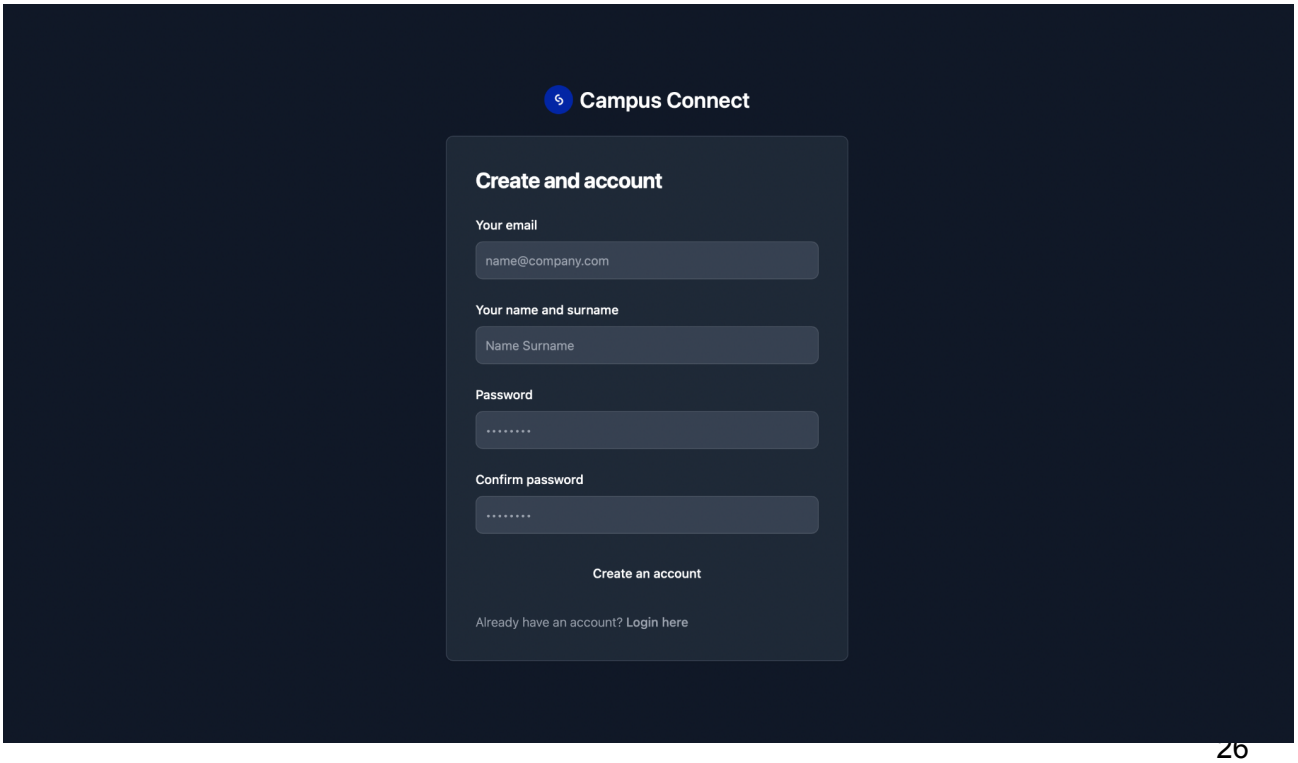


Fig. 12: Sign Up Page Including Sign Up Form

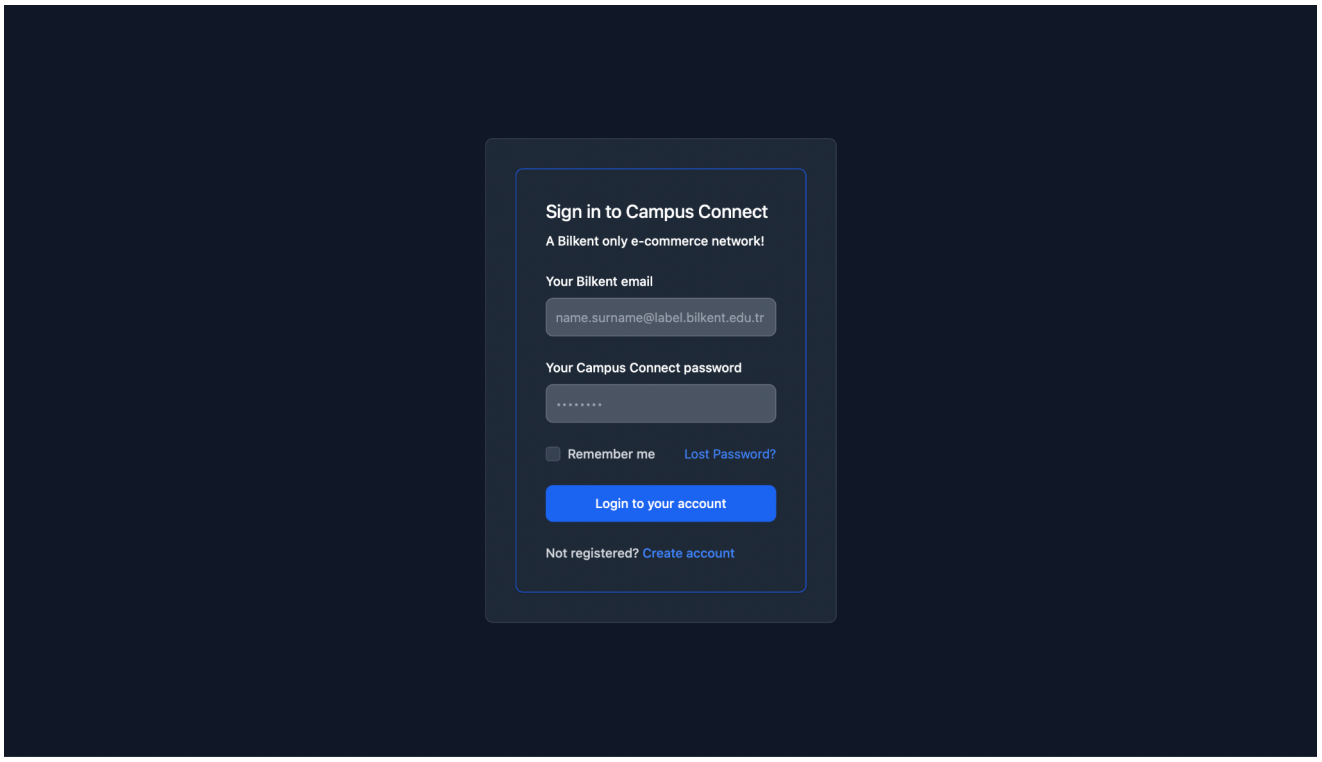


Fig. 13: Login Page Including Login Form

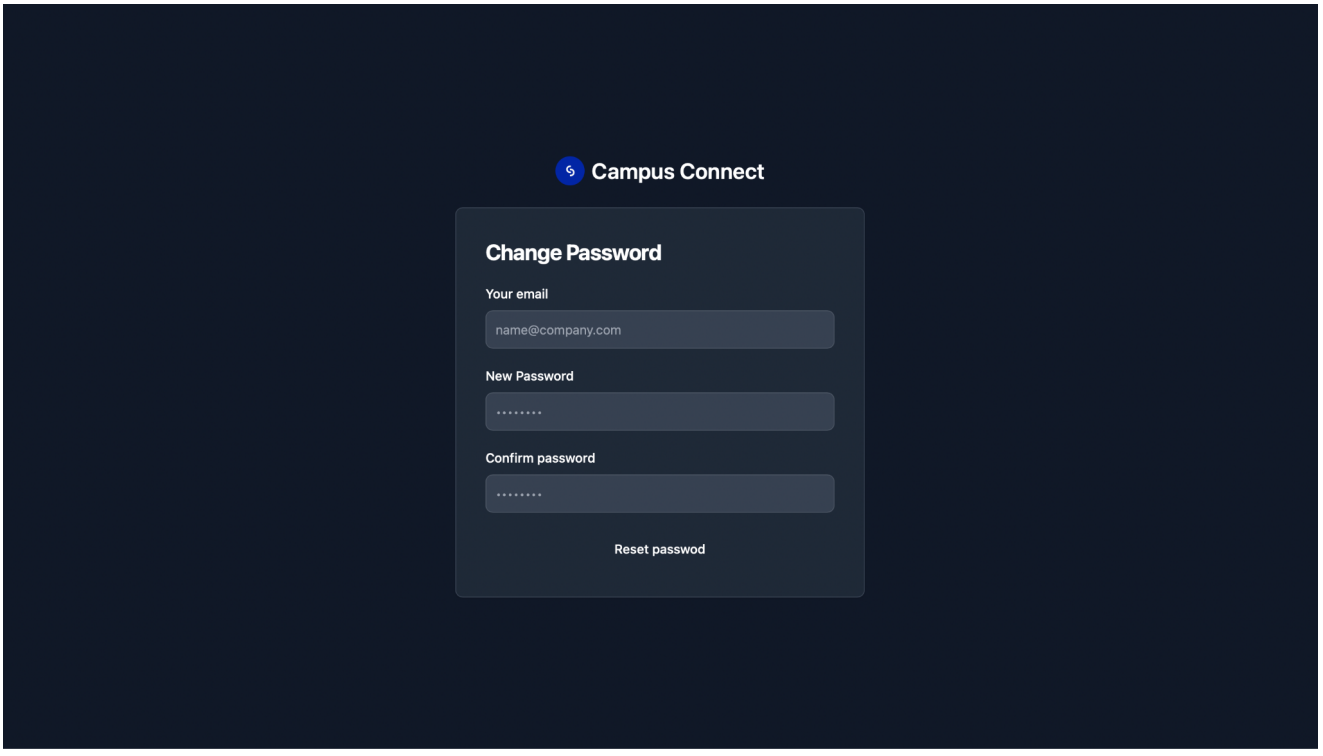


Fig. 14: Change Password Page Including Change Password Form

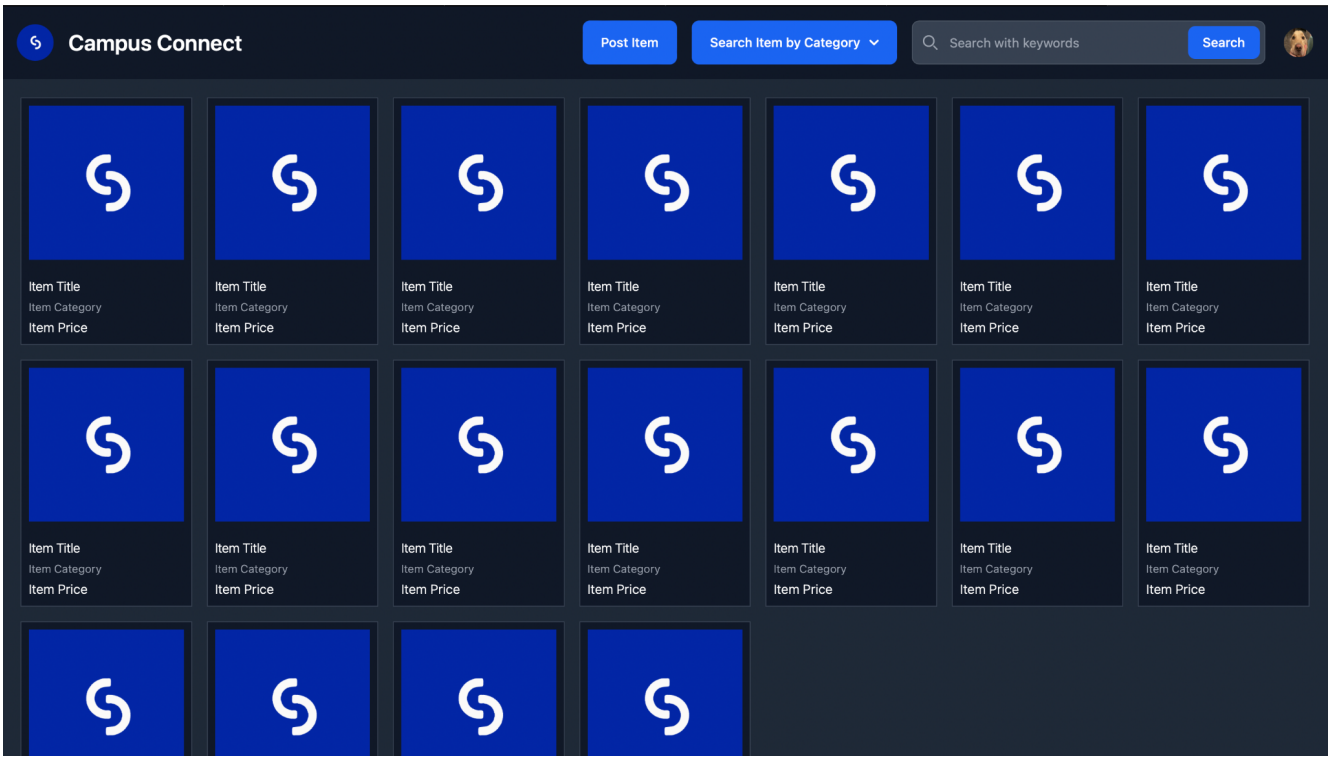


Fig. 15: Home Page Including Navbar and Latest Posts

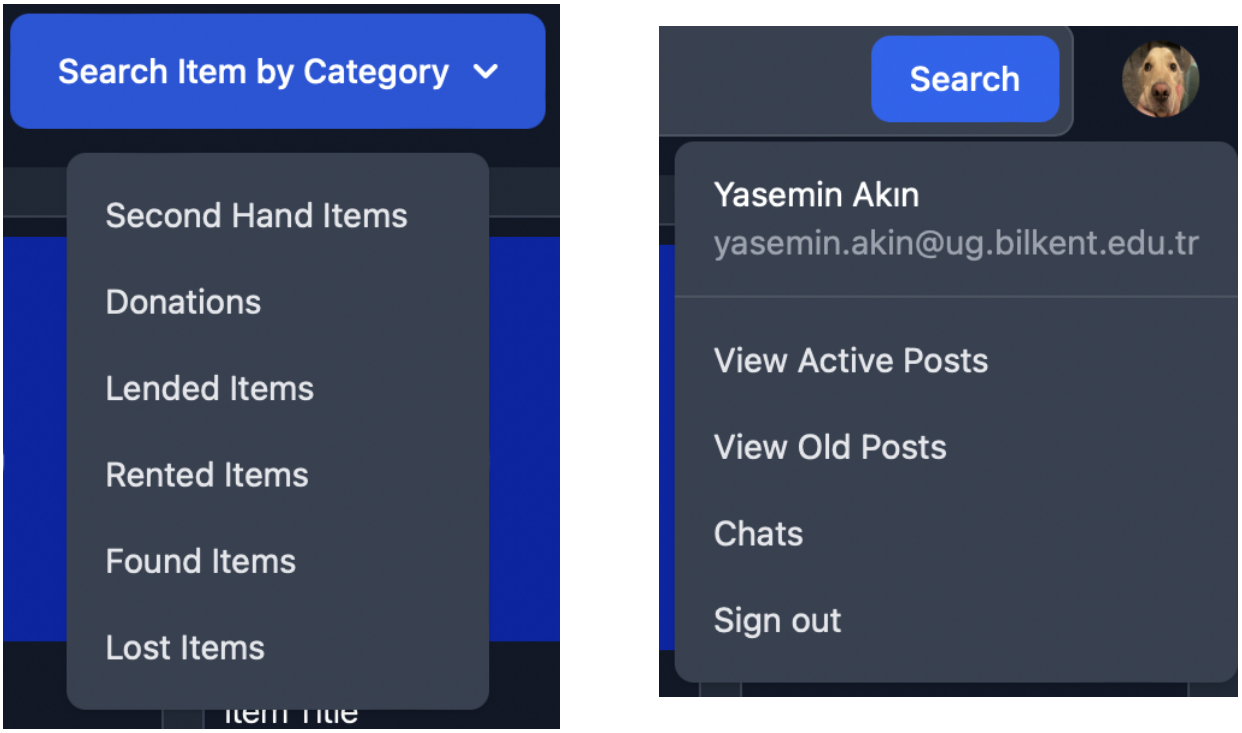


Fig. 16: Categories Dropdown Menu From Navbar, Category Anchors Lead Users to Category's Page and User Profile Dropdown Menu

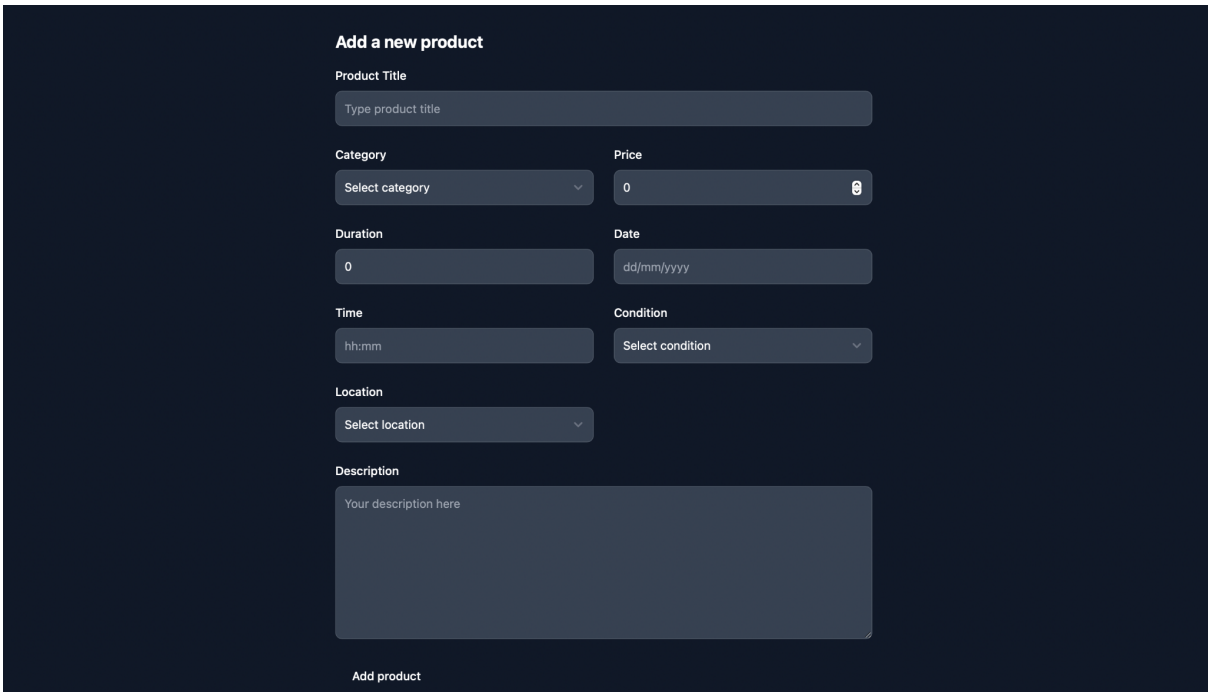


Fig. 17: Post Item Page Including Post Item Form

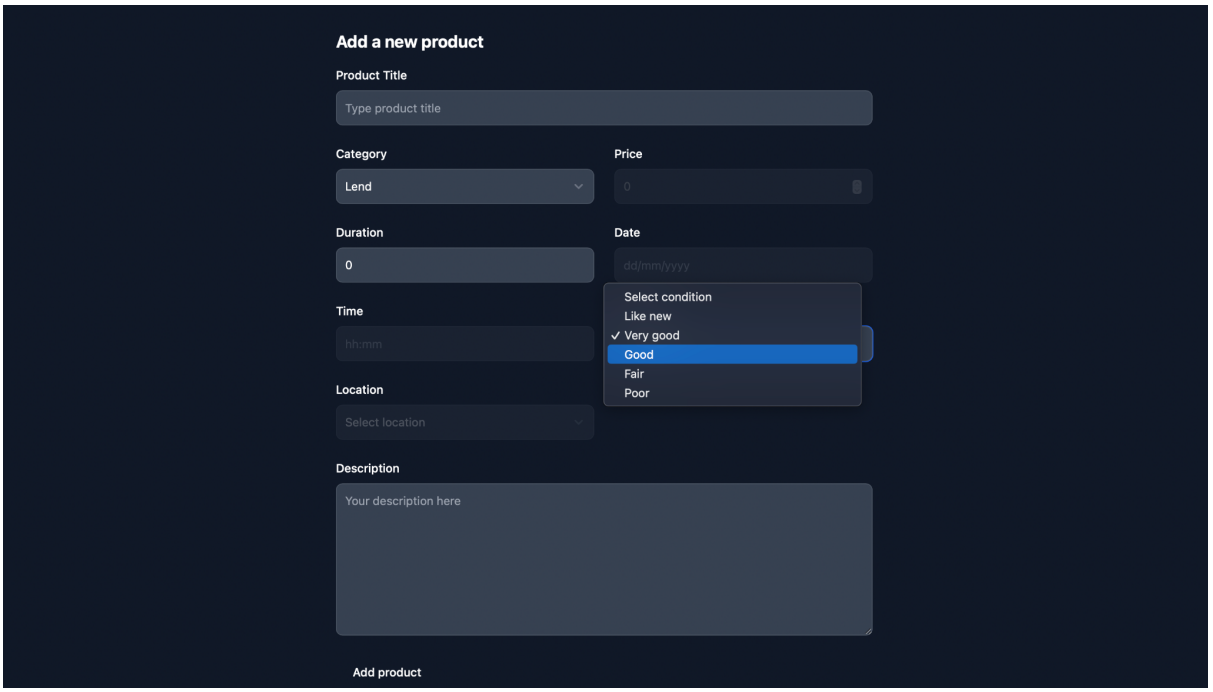


Fig. 18: Post Item Page Including Conditions Dropdown Menu Demonstrated (Also Demonstrates How Unrelated Form Fields Are Disabled According to Chosen Category)

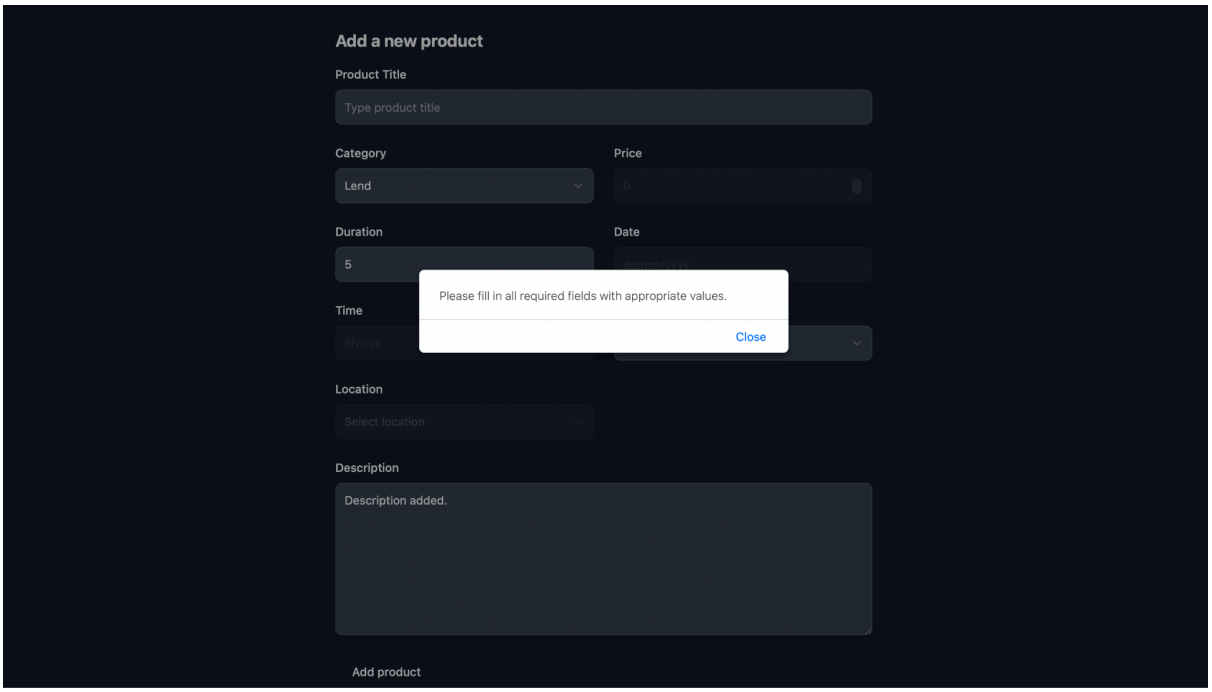


Fig. 19: Post Item Page Wrong or Incomplete Post Item Submission Alert

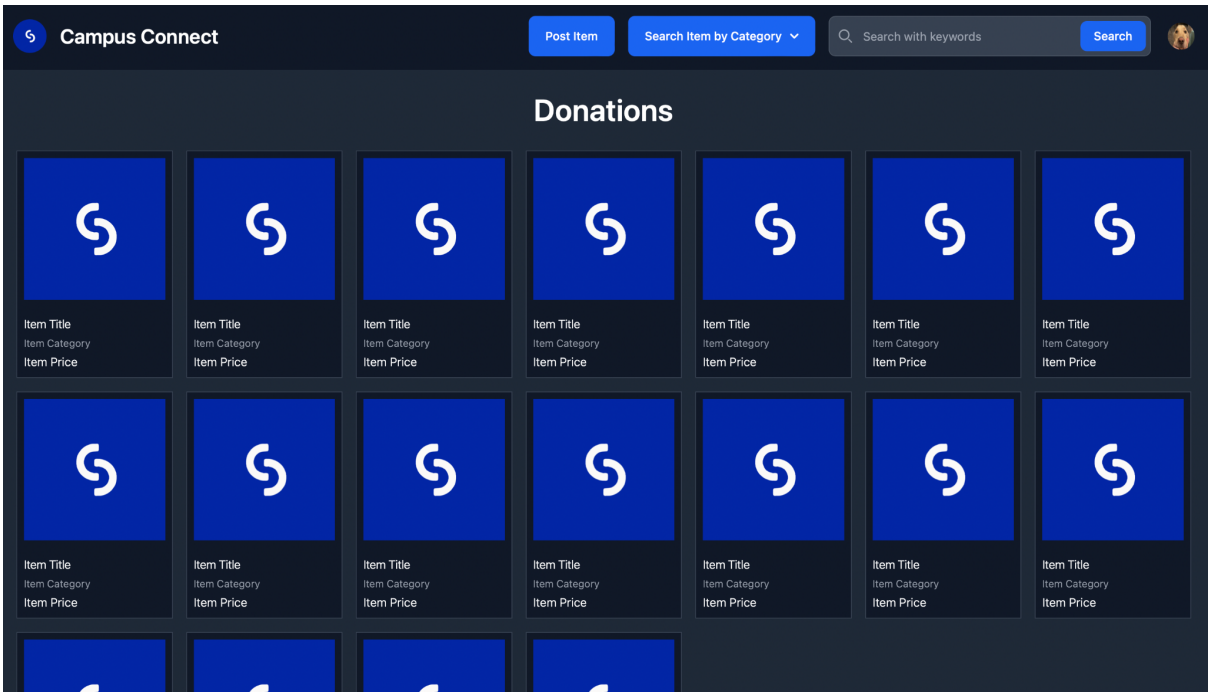


Fig. 20: Donations Category's Page

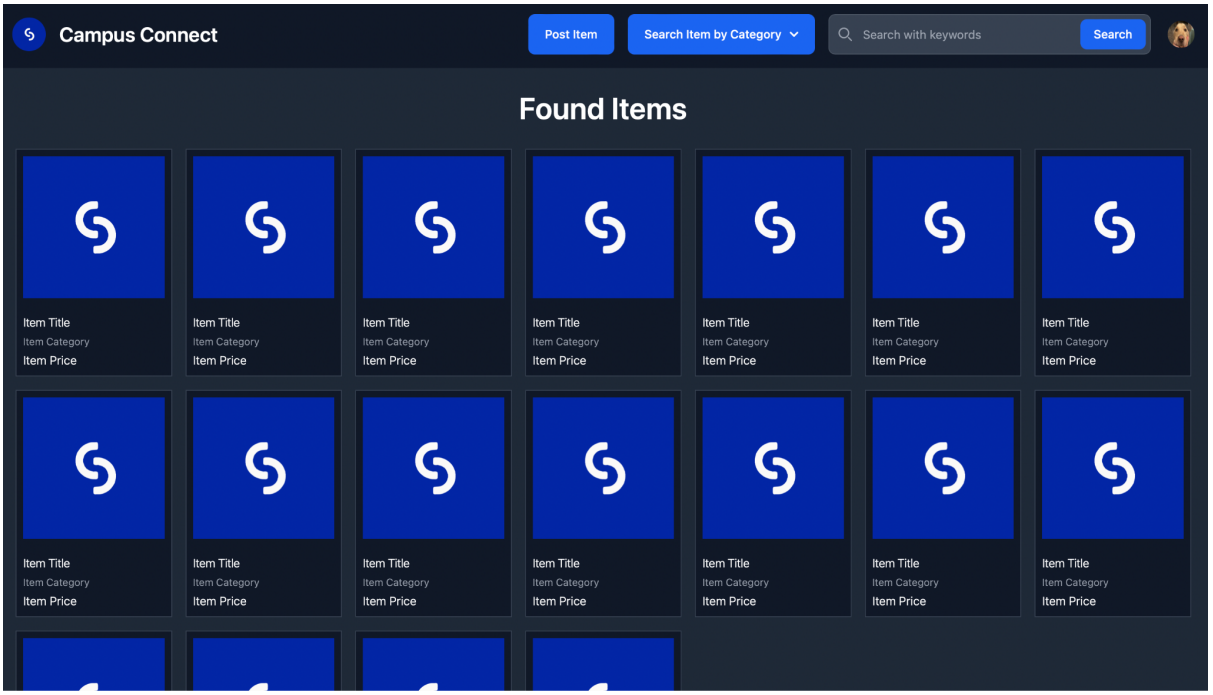


Fig. 21: Found Items Category's Page