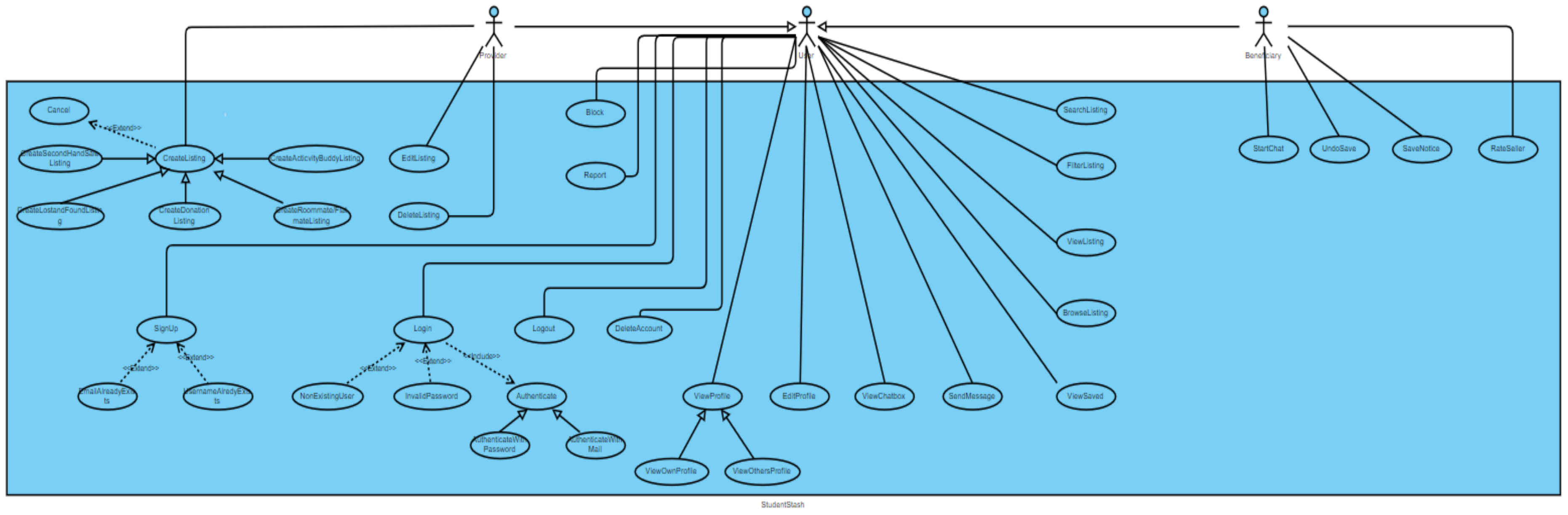
Use Case Diagram



Use case name: Login
Participating Actors: User
Entry Conditions:

- Opening the web application.

Exit Conditions:

- User signs in.
- Login process fails.
- User leaves the web app.

Flow of Events:

1. User decides to log in.
2. User fills the designated areas with her/his email and password
3. If the user enters wrong password,
    1. “Invalid Password” error shows up.
4. Else if the user entered a non-existing email,
    1. “Non-existing Email” error message shows up.
5. Else the user logs in successfully and the home page of the StudentStash opens up.

Use case name: SignUp
Participating Actors: User
Entry Conditions:

- Clicking the sign-up button on the login page.

Exit Conditions:

- User signs up.
- Sign up process fails.
- User leaves the web app.

Flow of Events:

1. User clicks the sign-up button on the login page.
2. If the two passwords entered during the sign-up mismatch,
    1. “Passwords mismatch” error shows up.
3. Else if the user entered an invalid Bilkent University mail during sign up,
    1. “Invalid Bilkent University Mail” error shows up.
4. Else if the Bilkent University mail entered during sign up already exists,
    1. “Bilkent University Mail already exists” error shows up.
5. Else if the username entered during sign up already exists,
    1. “Username is taken” error shows up.
6. Else the email verification email is sent
    1. The link is clicked.
    2. The user signs up successfully

Use case name: Log Out
Participating Actors: User
Entry Conditions:

- Clicking the logout button on their profile page.

- Clicking the logout button on the dropdown menu which is in the home page.

Exit Conditions:
- User logs out.

Flow of Events:
1. User clicks the logout button.
2. User logs out and the login page shows up.

Use case name: EmailAlreadyExists
Participating Actors: User
Entry Conditions:
- This use case extends the SignUp use case. It is initiated by the system whenever the email trying to be used already exists.

Exit Conditions:
- User turns back to SignUp use case.

Flow of Events:
1. User fills the designated area for the email with an already existing email.
2. “The email is already used.” warning is displayed.

Use case name: UsernameAlreadyExists
Participating Actors: User
Entry Conditions:
- This use case extends the SignUp use case. It is initiated by the system whenever the username trying to be used is already taken.

Exit Conditions:
- User turns back to SignUp use case.

Flow of Events:
1. User fills the designated area for the username with an already used username.
2. “The username is already taken.” warning is displayed.

Use case name: NonExistingUser
Participating Actors: User
Entry Conditions:
- This use case extends the Login use case. It is initiated by the system whenever the user tries to log in with an unregistered account mail.

Exit Conditions:
- User turns back to Login use case.

Flow of Events:
1. User fills the designated area for the email with an unregistered email.
2. “This user does not exist.” warning is displayed.

Use case name: InvalidPassword
Participating Actors: User

Entry Conditions:

- This use case extends the Log in use case. It is initiated by the system whenever the user tries to log in with incorrect password.

Exit Conditions:

- User turns back to Log in use case.

Flow of Events:

1. User fills the designated area for the password with an incorrect one.
2. “Invalid Password” warning is displayed.
3. User stays in the login page.

Use case name: DeleteAccount
Participating Actors: User
Entry Conditions:

- Clicking the delete account button on the profile page.

Exit Conditions:

- User turns back to sign up page.

Flow of Events:

1. User decides to delete their account and clicks the button.
2. A pop-up message, “Are you sure you want to delete this account?”, is displayed.
3. If the user wants to proceed, the account is deleted.
4. Else, the user turns back to their profile page.

Use case name: EditProfile
Participating Actors: User
Entry Conditions:

- Navigating to own profile and clicking to the edit button.

Exit Conditions:

- Clicking the save button or navigating to another page on the web application.

Flow of Events:

1. A participating user decides to edit their profile.
2. The user enters their profile and clicks the edit button.
3. The user makes the changes and clicks the save button.

Use case name: ViewChatbox
Participating Actors: User
Entry Conditions:

- Navigating to own profile and clicking to the ViewChatbox button.
- Clicking the ViewChatbox button on the dropdown menu which is on the home page.

Exit Conditions: Clicking the Exit button on the top right of the page.
Flow of Events:

    1. User decides to view their chatbox.
    2. User enters his/her chatbox through their profile.
    3. User turns back to his/her profile.

Use case name: SendMessage
Participating Actors: User
Entry Conditions:
- Clicking to StartChat button in the related listing and being directed to the chatbox.
- Entering the chatbox in the profile page.

Exit Conditions:
- Clicking to exit button on the top right corner.

Flow of Events:
1. User decides to send a message.
2. User types down a message to another user.
3. User sends the message by clicking to the Send button.

Use case name: ViewProfile
Participating Actors: User
Entry Conditions:
- Clicking the user name of the profile.

Exit Conditions:
- User turns back to the previous page.

Flow of Events:
1. User decides to view a profile.
2. User clicks the user name of the profile s/he wants to view.
3. User returns to the previous page.

Use case name: ViewOwnProfile
Participating Actors: Inherited from ViewProfile use case.
Entry Conditions:
- **Inherited** from ViewProfile use case.

Exit Conditions:
- **Inherited** from ViewProfile use case.

Flow of Events:
1. User decides to view his/her profile.
2. User clicks the user name of their profile.
3. User returns to the previous page.

Use case name: ViewOthersProfile
Participating Actors: Inherited from ViewProfile use case.
Entry Conditions:
- **Inherited** from ViewProfile use case.

Exit Conditions:

- **Inherited** from ViewProfile use case.

Flow of Events:
1. User decides to view another user's profile.
2. User clicks the user name of their profile.
3. User returns to the previous page.

Use case name: Authenticate
Participating Actors: User
Entry Conditions:
- Entering the Login page

Exit Conditions:
- Logging in to the account.

Flow of Events:
1. User enters the mail and password.
2. If the entered password is correct,
   - 2.1. User logs in successfully and the home page opens up.
3. Else if the entered password is incorrect,
   - 3.1. An email is sent for user to log in via a one time link.
   - 3.2. User logs in successfully and the home page opens up.

Use case name: AuthenticateWithMail
Participating Actors: Inherited from Authenticate.
Entry Conditions:
- **Inherited** from Authenticate use case.

Exit Conditions:
- **Inherited** from Authenticate use case.

Flow of Events:
1. User enters the mail and password.
2. Else if the entered password is incorrect,
   - 2.1. An email is sent for user to log in via a one time link.
   - 2.2. User logs in successfully and the home page opens up.

Use case name: AuthenticateWithPassword
Participating Actors: Inherited from Authenticate.
Entry Conditions:
- **Inherited** from Authenticate use case.

Exit Conditions:
- **Inherited** from Authenticate use case.

Flow of Events:
1. User enters the mail and password.
2. If the entered password is correct,
   - 2.1. User logs in successfully and the home page opens up.

Use case name: BrowseListing
Participating Actors: User
Entry Conditions:
- User must be in home page.

Exit Conditions:
- User leaves the page.

Flow of Events:
1. User enters the home page.
2. User scrolls through the listings unfiltered.
3. User leaves the page.

Use case name: ViewSaved
Participating Actors: User
Entry Conditions:
- User must be logged in.
- User must be in the saved listings page.

Exit Conditions:
- User leaves the page.

Flow of Events:
1. User logs in.
2. User goes to the saved listings page.
3. All the saved listings are displayed.
4. User leaves the page.

Use case name: RateSeller
Participating Actors: Beneficiary
Entry Conditions:
- User must be logged in.
- User must click on the rate button in the seller's profile

Exit Conditions:
- User closes the rate window, OR
- User submits the rating.

Flow of Events:
1. User logs in.
2. User goes to the seller's profile.
3. User clicks on the rate button.
4. Rate window opens up.
5. User selects a rating.
6. User submits the rating or closes the window.

Use case name: UndoSave
Participating Actors: Beneficiary
Entry Conditions:

- User must be logged in.
- User must have the listing saved.
- User must click the ‘undo save’ button

Exit Conditions:

- User successfully removes the listing from the saved listings.

Flow of Events:

1. User logs in.
2. User goes to the saved listing’s page.
3. User clicks the undo save button.
4. The listing gets removed from the saved listings.

Use case name: SaveNotice
Participating Actors: Beneficiary
Entry Conditions:

- User must be logged in.
- User doesn’t have the listing saved.
- User must click the save button in the listing’s page.

Exit Conditions:

- User successfully saves the listing.

Flow of Events:

1. User logs in.
2. User goes to the listing’s page.
3. User clicks the save button.
4. User successfully saves the listing.

Use case name: Block
Participating Actors: User
Entry Conditions:

- Entering another user’s profile and clicking the “Block” button.

Exit Conditions:

- User blocks the entered profile.

Flow of Events:

1. User decides to block another user’s profile.
2. User enters that profile and clicks the block button.
3. User is notified the profile has been blocked.

Use case name: Report
Participating Actors: User
Entry Conditions:
- Entering another user’s profile and clicking the “Report” button.

Exit Conditions:
- User reports the profile.

Flow of Events:
1. User decides to report another user’s profile.
2. User enters that profile and clicks the report button.
3. User is notified the profile has been reported.

Use case name: SearchListing
Participating Actors: User
Entry Conditions:
- User clicks and fills out the search bar on the main page.

Exit Conditions:
- User clicks the search icon or presses enter.

Flow of Events:
1. User clicks on the search bar.
2. User fills the search bar out with text to find a listing.
3. User clicks the search icon or presses enter to display the listings that fit the text provided.

Use case name: FilterListing
Participating Actors: User
Entry Conditions:
- User clicks the filter button on the main page.
- User clicks the filter button after searching for a listing.

Exit Conditions:
- User confirms the selected filters by pressing the “Filter” button.

Flow of Events:

1. User clicks the filter button to display the categories that listings can be filtered by.
2. User selects the filters that they want.
3. User confirms the filters they have chosen and clicks the “Filter” button to display the listings that fit the filters chosen.

Use case name: ViewListing
Participating Actors: User
Entry Conditions:
- User clicks on a listing on the main page.
- User clicks on a listing after searching.

- User clicks on a listing after filtering.

Exit Conditions:
- User clicks the back button when viewing a listing.
- User clicks the “Purchase” button after viewing a listing.
- User clicks the “Start chat with provider” button after viewing the listing.
- User clicks the “Sign-Up” button after viewing the listing.

Flow of Events:

1. User clicks on a listing of their choice.
2. User clicks the back, “Purchase”, “Start chat with provider” or “Sign-Up” button after viewing the listing.

Use case name: Start Chat
Participating Actors: Beneficiary
Entry Conditions:
- Starting a chat on a the post created for a listing (buy and sell, activities).

Exit Conditions:
- Buyer starts the chat.

Flow of Events:
1. Beneficiary enters the page for the post created for sales or activity notices, etc.
2. User clicks on the button "start chat"
3. User starts the chat with the person who has posted the notice and enquires about the listing.

Use case name: Create Listing
Participating Actors: Provider
Entry Conditions:
- Any user (either provider or beneficiary) can create a listing to post notices about roommate search, activities, second hand sales, etc. This automatically puts them in the position of the provider.

Exit Conditions:
- User clicks the cancel button.
- User successfully creates a listing.

Flow of Events:
1. Provider clicks on the button "Create Listing"
2. Creates a listing, enters in all the details about pricing, dates, etc.
3. Broadcasts the listing and listing appears on the feed.

Use case name: Create second-hand sales listing
Participating Actors: Provider
Entry Conditions:
- User clicks on the creating listing button on the menu
- User labels the created listing as "second hand sales"

Exit Conditions:
- User clicks the cancel button.
- User successfully creates a listing.

Flow of Events:
1. Provider clicks on the button "Create Listing"
2. Creates a listing, enters in all the details about pricing, etc.
3. Broadcasts the listing and listing appears on the feed.

Use case name: Create donation listing
Participating Actors: Provider
Entry Conditions:

- User clicks on the creating listing button on the menu
- User labels the created listing as “donation”

Exit Conditions:
- User clicks the cancel button.
- User successfully creates a listing.

Flow of Events:
1. Provider clicks on the button “Create Listing”
2. Creates a listing, enters in all the details about the item to be donated.
3. Broadcasts the listing and listing appears on the feed.

Use case name: Create activity-buddy listing
Participating Actors: Provider
Entry Conditions:
- User clicks on the creating listing button on the menu
- User labels the created listing as “Activity”

Exit Conditions:
- User clicks the cancel button.
- User successfully creates a listing.

Flow of Events:
1. Provider clicks on the button “Create Listing”
2. Creates a listing, enters in all the details about dates, etc.
3. Broadcasts the listing and listing appears on the feed.

Use case name: Create roommate/flatmate listing
Participating Actors: Provider
Entry Conditions:
- User clicks on the creating listing button on the menu
- User labels the created listing as “roommate/flatmate search”

Exit Conditions:
- User clicks the cancel button.
- User successfully creates a listing.

Flow of Events:
1. Provider clicks on the button “Create Listing”
2. Creates a listing, enters in all the details about location, pricing, etc.
3. Broadcasts the listing and listing appears on the feed.

Use case name: Create Lost and Found listing
Participating Actors: Provider
Entry Conditions:
- User clicks on the creating listing button on the menu
- User labels the created listing as “Lost and Found Items”

Exit Conditions:
- User clicks the cancel button.
- User successfully creates a listing.

Flow of Events:

1. Provider clicks on the button “Create Listing”
2. Creates a listing, enters in all the details about the lost or found items, photographs, etc.
3. Broadcasts the listing and listing appears on the feed.

Use case name: EditListing
Participating Actors: Provider
Entry Conditions:

- User clicks on the edit listing button on the listing.

Exit Conditions:

- User clicks the cancel button.
- User successfully edits the listing.

Flow of Events:

1. User clicks on the edit button.
2. If user clicks on the cancel button,
    2.1. user returns back to the listing page.
3. Else if the user clicks on the save button,
    3.1. Listing gets edited successfully.

Use case name: Delete Listing
Participating Actors: Provider
Entry Conditions:

- User clicks on the delete listing button on the listing and erases it off the feed.

Exit Conditions:

- User clicks the cancel button.
- User successfully edits the listing.

Flow of Events:

1. User clicks on the delete button.
2. If user clicks on the cancel button,
    a. user returns back to the listing page.
3. Else if the user clicks on the “yes, I am sure” button,
4. Listing gets deleted successfully.

Nonfunctional Requirements

- Deleted listings should not be visible to users. (Usability Requirement)
- User-friendly interface (Usability Requirement):
    - While viewing a listing, all features of that listing should be clear and easy to spot.
    - Colors and panels used in the UI should be concise and minimal.
- Users can only register with a Bilkent University mail. (Safety Requirement)
- All of the staff and students of Bilkent University should be able to use the web application with their Bilkent University mail. (Usability Requirement)
- Project should be implemented in an Object-Oriented manner. (Constraint)
- Project should be designed as a web application. (Constraint)

Tech Stack

MERN:

- MongoDB for database
- Express(.js) for web framework
- React(.js) for JavaScript framework
- Node(.js) for the JavaScript web server

We chose MERN for its usability and efficiency. It will allow us to use JavaScript throughout all of our application stack which will let us reuse our code. It will make our project more flexible and accelerate the development speed.