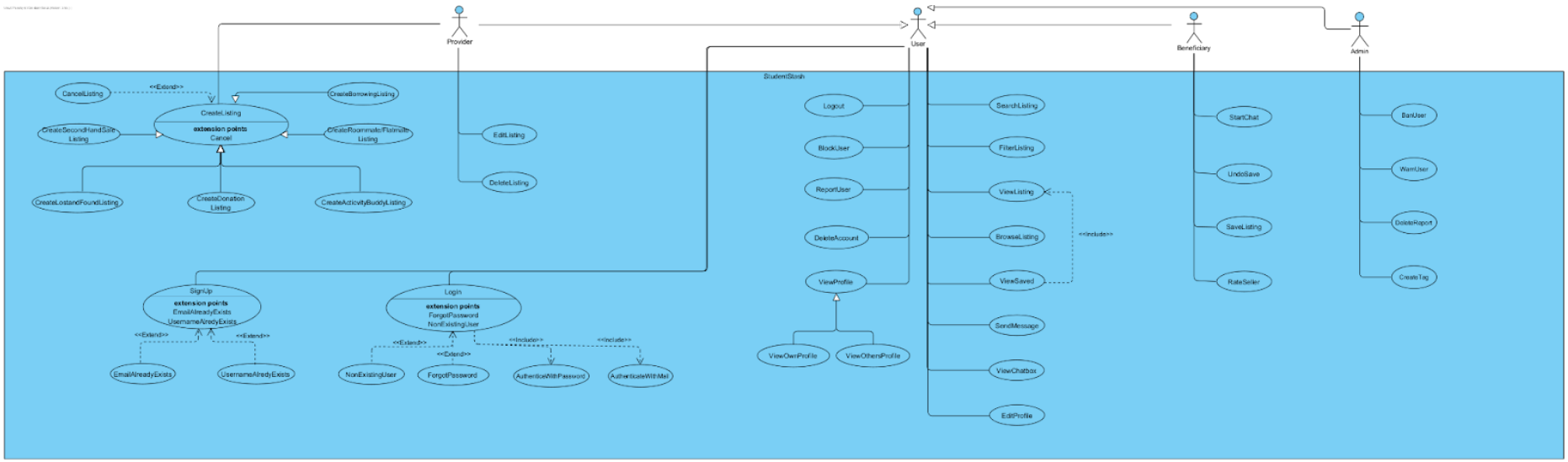
CS319 - Object Oriented Software Engineering
Project Analysis Report
Section 001
Team 1

Student Stash

Alp Batu Aksan - 22103246
Çağan Tuncer - 22102018
Defne Gürbüz - 22103295
İrem Hafızoğlu - 22101848
Ümmügülsüm Sümer - 22103772
Ghazal Amirrashed - 22001446

Contents

1. Use Case Diagram



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

- Opening the web application.

Exit Conditions:

- The user signs in.
- The login process fails.
- The user leaves the web app.

Flow of Events:

1. The user decides to log in.
2. The user fills the designated areas with their email and password.
3. The user logs in successfully, and the home page of the StudentStash opens up.

Alternative Flows:

1.
    1. The user decides to log in.
    2. The user fills in the designated areas with their email and password.
    3. The user enters the wrong password.
    4. The "Invalid Password" error shows up.

2.
    1. The user decides to log in.
    2. The user fills the designated areas with their email and password.
    3. The user enters a non-existing email.
    4. The "Non-existing Email" error message shows up.

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

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

Exit Conditions:

- The user signs up.
- The sign-up process fails.
- The user leaves the web app.

Flow of Events:

1. The user clicks the sign-up button on the login page.
2. The user fills the designated areas with their email, password, and user name.
3. The email verification email is sent.
4. The link is clicked.
5. The user signs up successfully.

Alternative Flows:

1.
    1. The user clicks the sign-up button on the login page.
    2. The user fills the designated areas with their email, password, and user name.
    3. The two passwords entered during the sign-up mismatch.
    4. "Passwords mismatch" error shows up.

2.
    1. The user clicks the sign-up button on the login page.
    2. The user fills the designated areas with their email, password, and user name.
    3. The user enters an invalid Bilkent University mail during sign-up.
    4. “Invalid Bilkent University Mail” error shows up.

3.
    1. The user clicks the sign-up button on the login page.
    2. The user fills the designated areas with their email, password, and user name.
    3. The Bilkent University mail entered during sign-up already exists.
    4. The “Bilkent University Mail already exists” error shows up.

4.
    1. The user clicks the sign-up button on the login page.
    2. The user fills the designated areas with their email, password, and user name.
    3. The username entered during sign-up already exists.
    4. The “Username is taken” error shows up.

Use case name: LogOut
Participating Actors: User
Entry Conditions:
- Clicking the logout button on their profile page.
- Clicking the logout button on the dropdown menu, which is on the home page.

Exit Conditions:
- The user logs out.

Flow of Events:
1. The user clicks the logout button.
2. The 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:
- The user turns back to the SignUp use case.

Flow of Events:
1. The 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:

- The user turns back to the SignUp use case.

Flow of Events:

1. The 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:

- The user turns back to the Login use case.

Flow of Events:

1. The user fills the designated area for the email with an unregistered email.
2. "This user does not exist." warning is displayed.

Use case name: ForgotPassword
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 an incorrect password.

Exit Conditions:

- The user turns back to Log in use case.

Flow of Events:

1. The user fills the designated area for the password with an incorrect one.
2. "Invalid Password" warning is displayed.
3. The user stays on the login page.

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

- Clicking the delete account button on the profile page.

Exit Conditions:

- The user turns back to the sign-up page.

Flow of Events:

1. The 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. The user clicks "yes," and the account gets deleted successfully.

Alternative Flows:

1. The 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. The user clicks “no” and turns back to their profile page.

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

- Navigating to your own profile and clicking on 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 your own profile and clicking on 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. The user decides to view their chatbox.
2. The user enters their chatbox through their profile.
3. The user turns back to their profile.

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

- Clicking on the StartChat button in the related listing and being directed to the chatbox.
- Entering the chatbox on the profile page.

Exit Conditions:

- Clicking the exit button on the top right corner.

Flow of Events:

1. The user decides to send a message.
2. The user types down a message to another user.
3. The user sends the message by clicking on the Send button.

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

- Clicking the user name of the profile.

Exit Conditions:

- The user turns back to the previous page.

Flow of Events:

1. The user decides to view a profile.
2. The user clicks the user name of the profile s/he wants to view.
3. The 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. The user decides to view their profile.
2. The user clicks the user name of their profile.
3. The 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. The user decides to view another user's profile.
2. The user clicks the user name of their profile.
3. The user returns to the previous page.

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

- Entering the Login page.

Exit Conditions:

- Logging in to the account.

Flow of Events:

1. The user enters their mail and password.
2. When the entered password is incorrect, the message "You have entered wrong password" is shown on the screen. An email is sent for the user to log in via a one-time link.
3. The user logs in through the sent link successfully, and the home page opens up.

Use case name: AuthenticateWithPassword
Participating Actors: User.
Entry Conditions:

- Entering the Login page.

Exit Conditions:

- Logging in to the account.

Flow of Events:

1. The user enters the mail and password correctly.
2. If the entered password is correct,
3. The user logs in successfully, and the home page opens up.

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

- The user must be on the home page.
- The user must be logged in.

Exit Conditions:

- The user leaves the page.

Flow of Events:

1. The user enters the home page.
2. The user scrolls through the listings unfiltered.
3. The user leaves the page.

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

- The user must be logged in.
- The user must be on the saved listings page.

Exit Conditions:

- The user leaves the page.

Flow of Events:

1. The user logs in.
2. The user goes to the saved listings page.
3. All the saved listings are displayed.
4. The user leaves the page.

Alternative Flows:

1.
    1. The user logs in.
    2. The user goes to the saved listings page.
    3. All the saved listings are displayed.
    4. The user clicks on the listing they want to view.
    5. The user uses the ViewListing use class.

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

- The user must be logged in.
- The user must click on the rating button in the seller's profile

Exit Conditions:

- The user closes the rate window.
- The user submits the rating.

Flow of Events:

1. The user logs in.
2. The user goes to the seller's profile.
3. The user clicks on the rate button.
4. The rate window opens up.
5. The user selects a rating.
6. The user submits the rating or closes the window.

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

- The user must be logged in.
- The user must have the listing saved.
- The user must click the 'undo save' button

Exit Conditions:

- The user successfully removes the listing from the saved listings.

Flow of Events:

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

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

- The user must be logged in.
- The user doesn't have the listing saved.
- The user must click the save button on the listing's page.

Exit Conditions:

- The user successfully saves the listing.

Flow of Events:

1. The user logs in.
2. The user goes to the listing's page.
3. The user clicks the save button.
4. The user successfully saves the listing.

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

- Entering another user's profile and clicking the "Block" button.

Exit Conditions:

- The user blocks the entered profile.

Flow of Events:

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

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

- Entering another user's profile and clicking the "Report" button.

Exit Conditions:

- The user reports the profile.

Flow of Events:

1. The user decides to report another user's profile.
2. The user enters that profile and clicks the report button.
3. The user is notified the profile has been reported.

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

- The user clicks and fills out the search bar on the main page.

Exit Conditions:

- The user clicks the search icon or presses enter.

Flow of Events:

1. The user clicks on the search bar.
2. The user fills the search bar with text to find a listing.
3. The 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:

- The user clicks the filter button on the main page.
- The user clicks the filter button after searching for a listing.

Exit Conditions:

- The user confirms the selected filters by pressing the "Filter" button.

Flow of Events:

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

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

- The user clicks on a listing on the main page.
- The user clicks on a listing after searching.
- The user clicks on a listing after filtering.
- User clicks on a listing in their saved listings

Exit Conditions:

- The user clicks the back button when viewing a listing.
- The user clicks the "Start chat with provider" button after viewing the listing.

Flow of Events:

1. User clicks on a listing of their choice.
2. The user clicks the back or "Start a chat with provider" button.

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

- Starting a chat on the post created for a listing (buy and sell, activities).

Exit Conditions:

- The buyer starts the chat.

Flow of Events:

1. Beneficiary enters the page for the post created for sales or activity notices, etc.
2. The user clicks on the button "Start Chat."
3. The user starts the chat with the person who has posted the notice and enquires about the listing.

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

- Any user (provider or beneficiary) can create a listing to post notices about roommate searches, activities, second-hand sales, etc. This automatically puts them in the position of the provider.

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

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

Use case name: CreateSecondHandSaleListing
Participating Actors: Inherited from CreateListing use case.
Entry Conditions:

- The user clicks on the "Create Listing" button on the menu.
- The user labels the created listing as "Second-Hand Sale."

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

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

Use case name: CreateDonationListing
Participating Actors: Inherited from CreateListing use case.
Entry Conditions:

- The user clicks on the "Create Listing" button on the menu.
- The user labels the created listing as "Donation."

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

1. Provider clicks on the button "Create Listing."
2. Creates a listing and enters all the details about the item to be donated.
3. Broadcasts the listing, and the listing appears on the feed.

Use case name: CreateActivityBuddyListing
Participating Actors: Inherited from CreateListing use case.
Entry Conditions:

- The user clicks on the "Create Listing" button on the menu.
- The user labels the created listing as "Activity."

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

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

Use case name: CreateRoommate/FlatmateListing
Participating Actors: Inherited from CreateListing use case.
Entry Conditions:

- The user clicks on the "Create Listing" button on the menu
- The user labels the created listing as "Roommate/Flatmate."

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

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

Use case name: CreateLostandFoundListing
Participating Actors: Inherited from CreateListing use case.
Entry Conditions:

- The user clicks on the "Create Listing" button on the menu
- The user labels the created listing as "Lost and Found Items."

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

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

Use case name: CreateBorrowingListing
Participating Actors: Inherited from CreateListing use case.
Entry Conditions:

- The user clicks on the “Create Listing” button on the menu
- The user labels the created listing as “Borrowing.”

Exit Conditions:

- The user clicks the cancel button.
- The user successfully creates a listing.

Flow of Events:

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

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

- The user must be on one of their listings.

Exit Conditions:

- The user cancels the listing successfully.

Flow of Events:

1. Provider clicks on the button “Cancel Listing.”
2. “Are you sure you want to cancel this listing?” message shows on the screen.
3. User clicks “Yes” and the listing gets cancelled.

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

- The user clicks on the edit listing button on the listing.

Exit Conditions:

- The user clicks the cancel button.
- The user successfully edits the listing.

Flow of Events:

1. The user clicks on the edit button.
2. The user makes the desired changes and clicks the “Save” button.
3. The listing gets edited successfully.

Use case name: DeleteListing
Participating Actors: Provider, Admin
Entry Conditions:

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

Exit Conditions:

- The user clicks the cancel button.
- The user successfully deletes the listing.

Flow of Events:

1. The user clicks on the delete button.
2. The "Are you sure you want to delete this listing?" message appears on the screen.
3. The user clicks on the "Yes" button.
4. The listing gets deleted successfully.

Alternative Flows:

1.
    1. The user clicks on the delete button.
    2. The "Are you sure you want to delete this listing?" message appears on
    3. The user clicks on the "No" button.
    4. The user turns back to the previous page.

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

- User must type something in the search bar.
- User must click the search button.

Exit Conditions:

- User leaves the page.

Flow of Events:

1. User types something in the search button.
2. User clicks the search button.
3. User sees the listings that include the typed string in the search bar.
4. 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:

- Users 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 or clicks on a saved listing.

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.
- 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: BanUser
Participating Actors: Admin
Entry Conditions:

- Admin must be logged in.
- The user who will be banned should be warned thrice.

Exit Conditions:

- Admin successfully bans the user from the application.

Flow of Events:

1. Admin clicks to "Ban User" button.
2. "Are your sure you want to ban this user?" message shows up.
3. Admin clicks "Yes" button and the user gets banned from the application.

Use case name: WarnUser
Participating Actors: Admin
Entry Conditions:

- Admin must be logged in.
- The user who will be warned should have been reported.

Exit Conditions:

- Admin warns the user.

Flow of Events:

1. Admin checks the report and decides to warn the user.
2. Admin clicks to “Warn User” button.
3. “Are your sure you want to warn this user?” message shows up.
4. Admin clicks “Yes” button and the user gets warned.

Use case name: DeleteReport
Participating Actors: Admin
Entry Conditions:

- Admin must be logged in.

Exit Conditions:

- Admin deletes the report.

Flow of Events:

1. Admin checks the report and decides to delete the report.
2. Admin clicks to “Delete Report” button.
3. Report gets deleted.

Use case name: CreateTag
Participating Actors: Admin
Entry Conditions:

- Admin must be logged in.

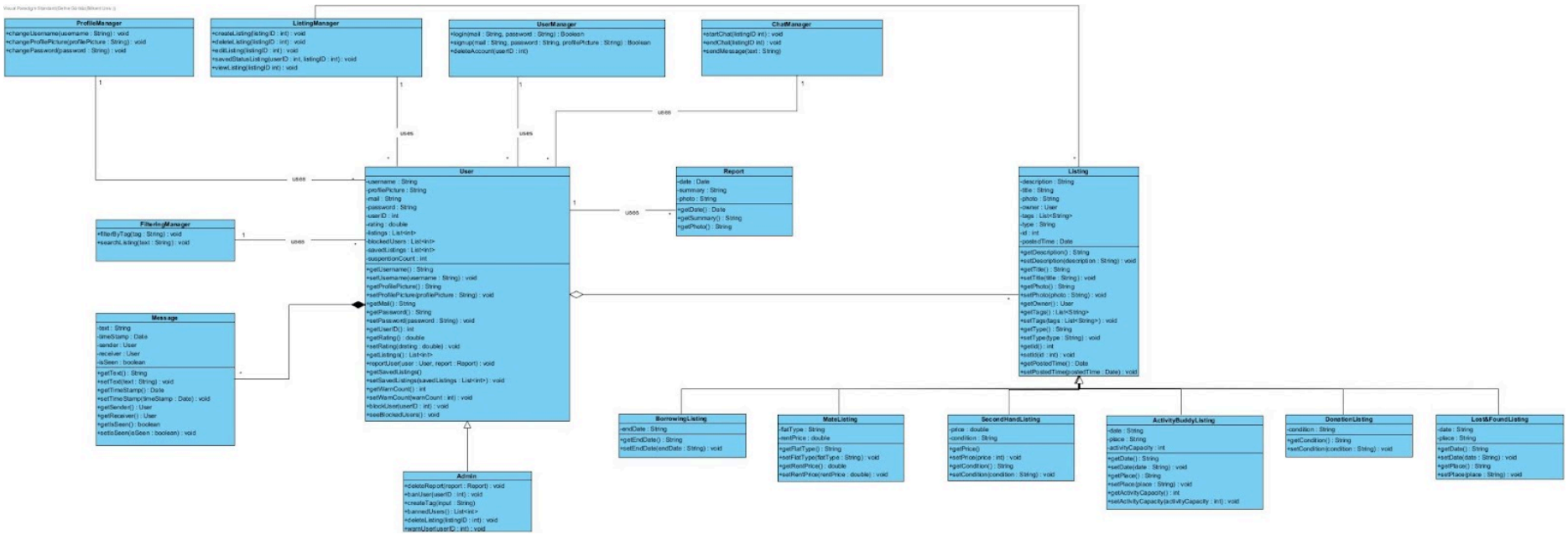
Exit Conditions:

- Admin adds the tag.

Flow of Events:

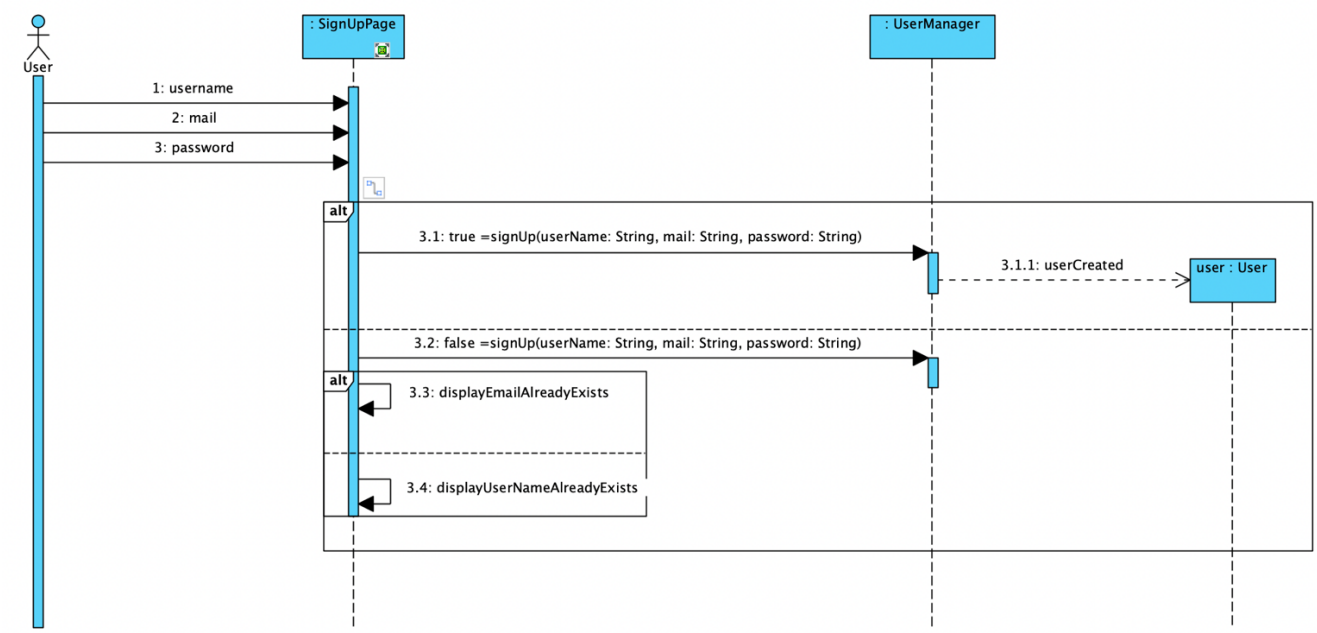
1. Admin clicks to the “Create Tag” button.
2. Admin enters the name of the tag.
3. Admin clicks to “Add Tag” button.
4. Tag gets added.

2. Class Diagram

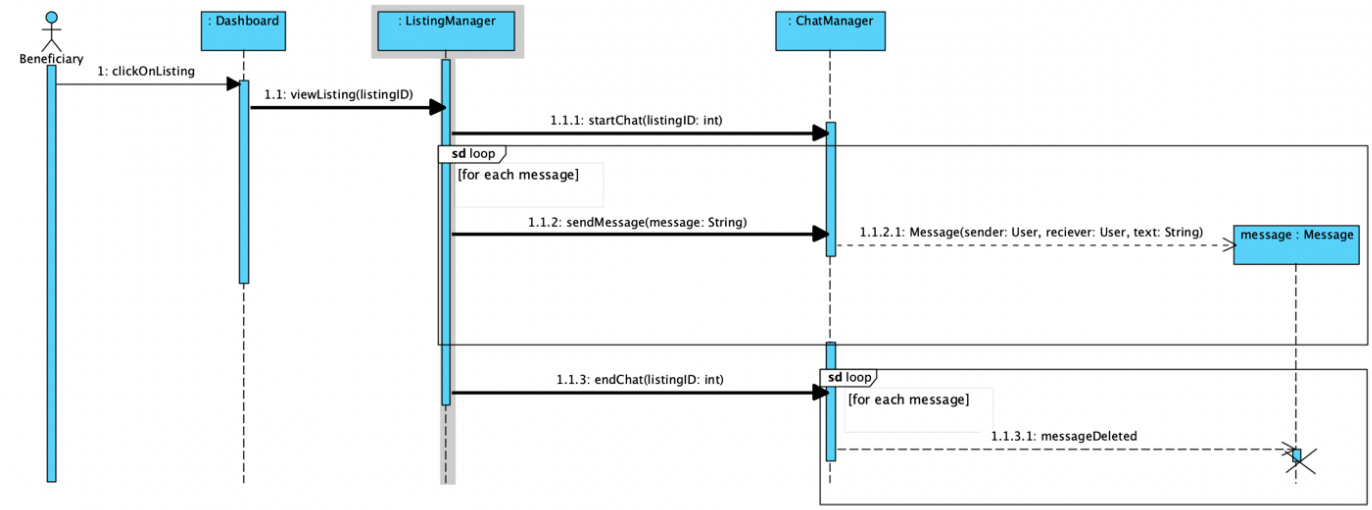


3. Sequence Diagrams

3.1. Sign-Up Sequence Diagram

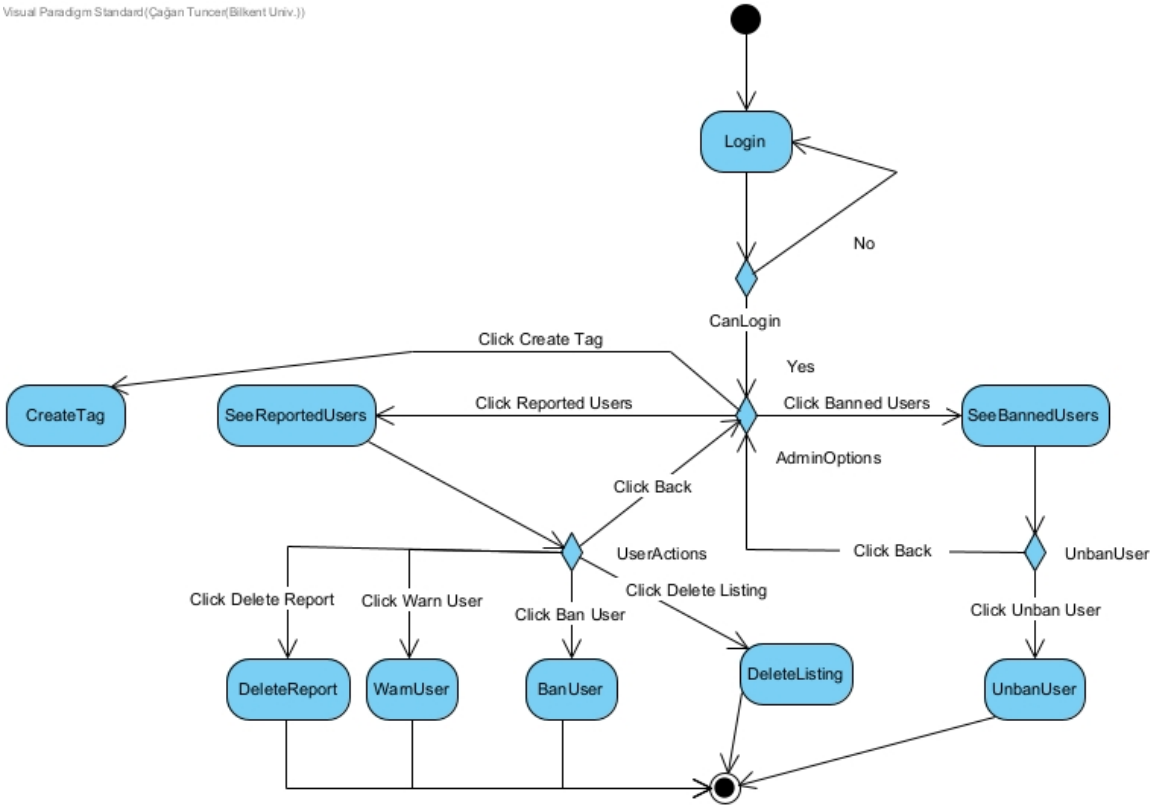


3.2. Chatting Sequence Diagram

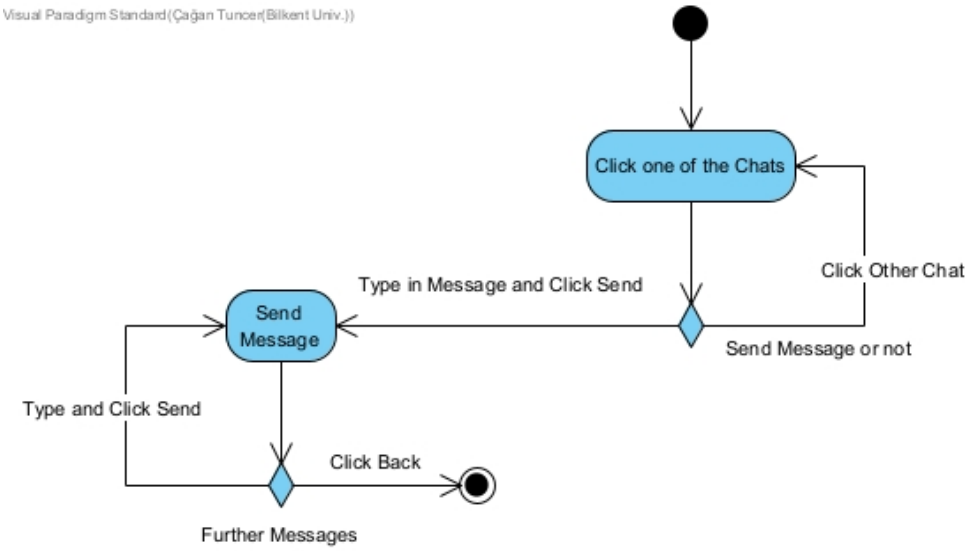


4. Activity Diagrams

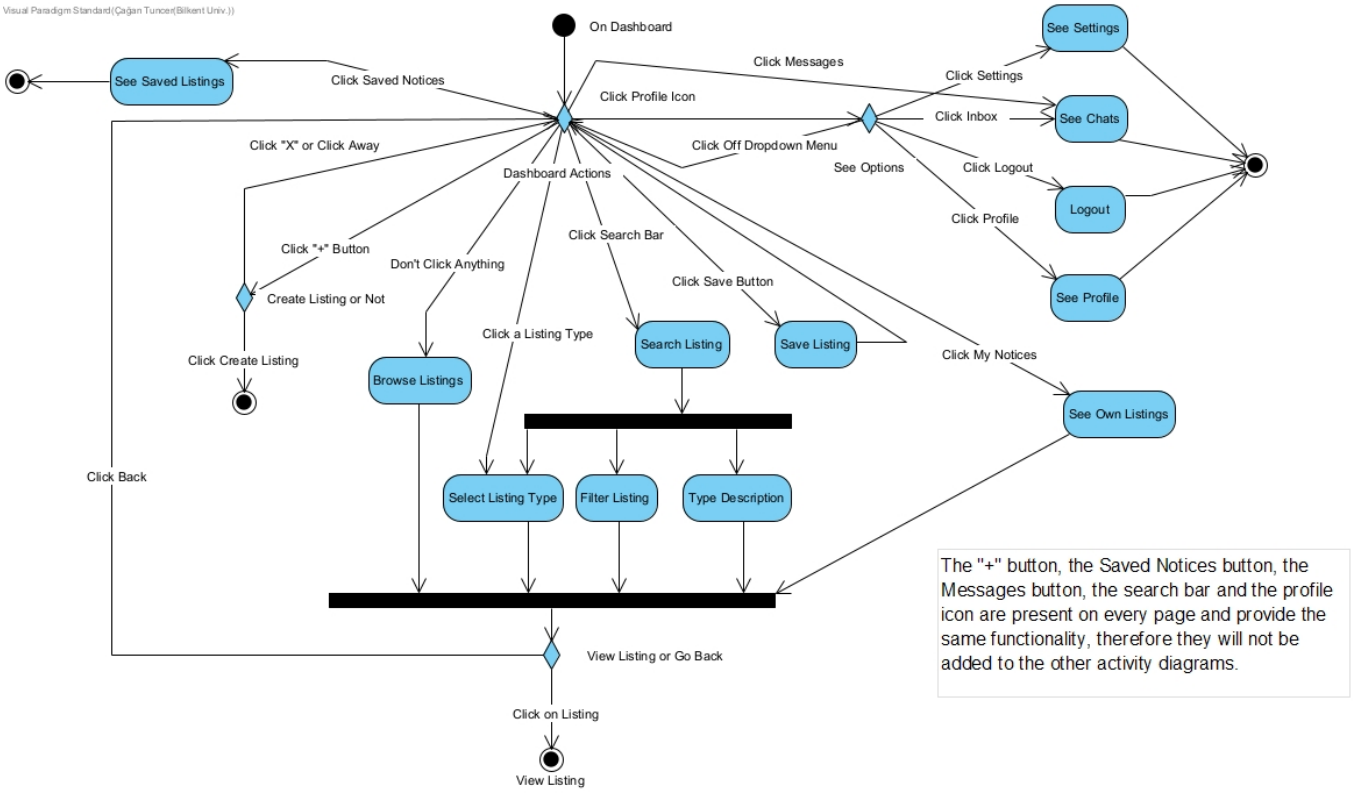
4.1. Admin Activity Diagram



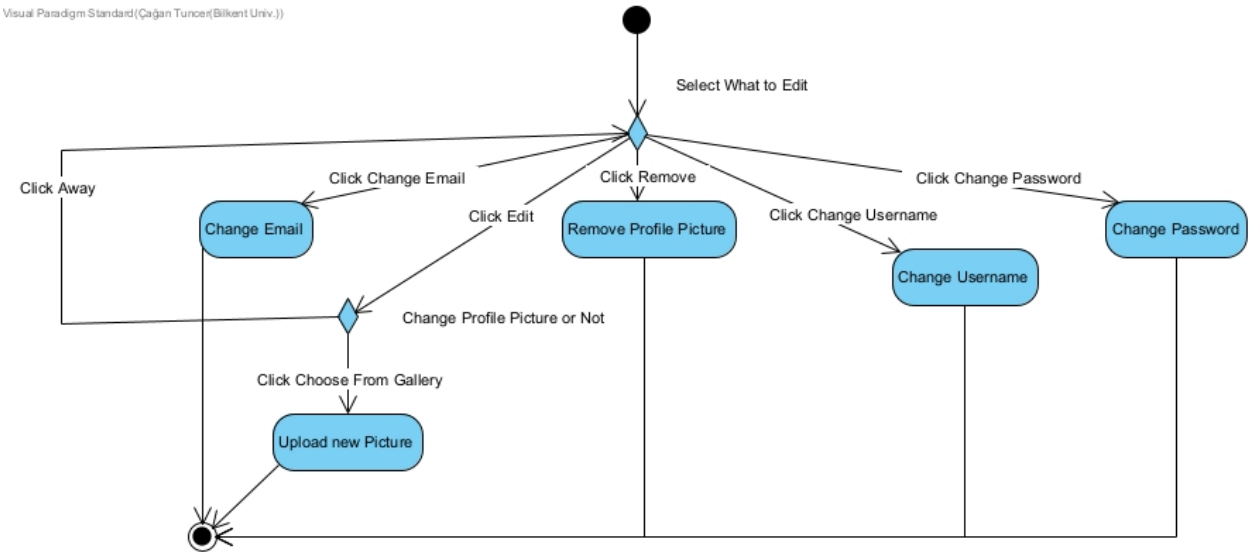
4.2. Chat Activity Diagram



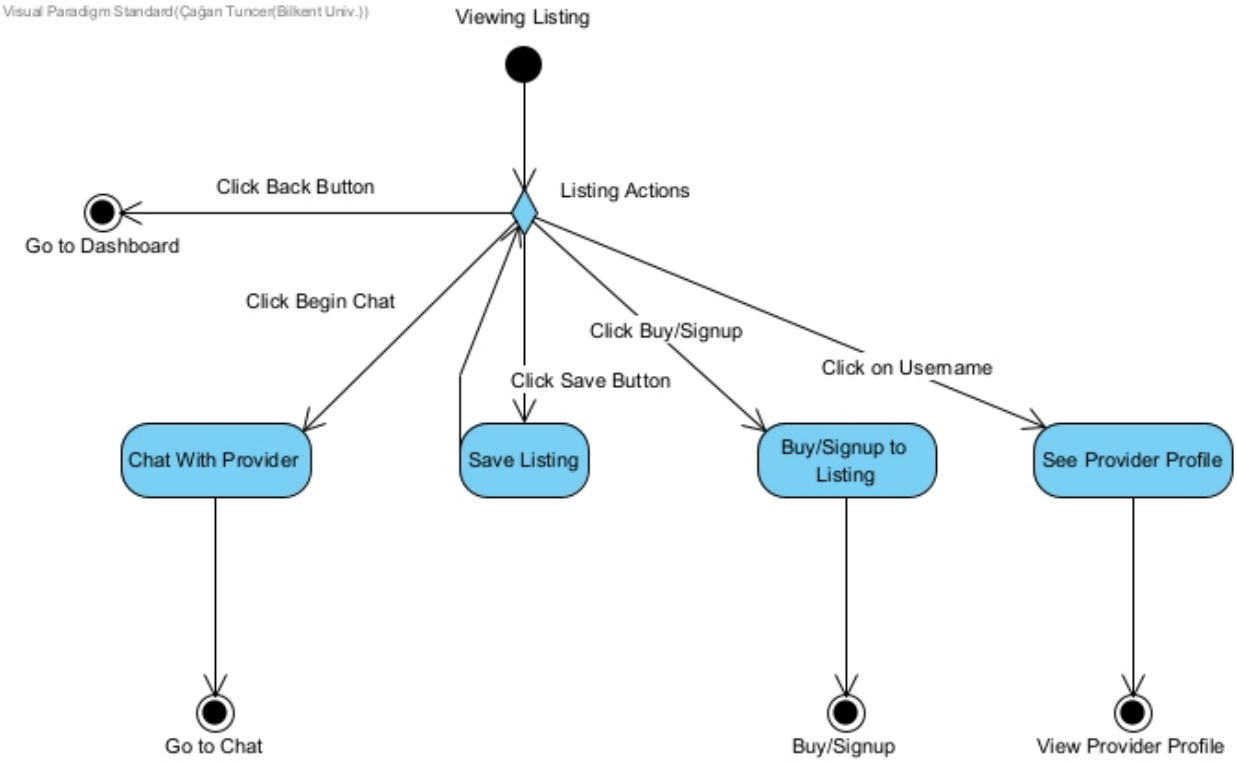
4.3. Dashboard Activity Diagram



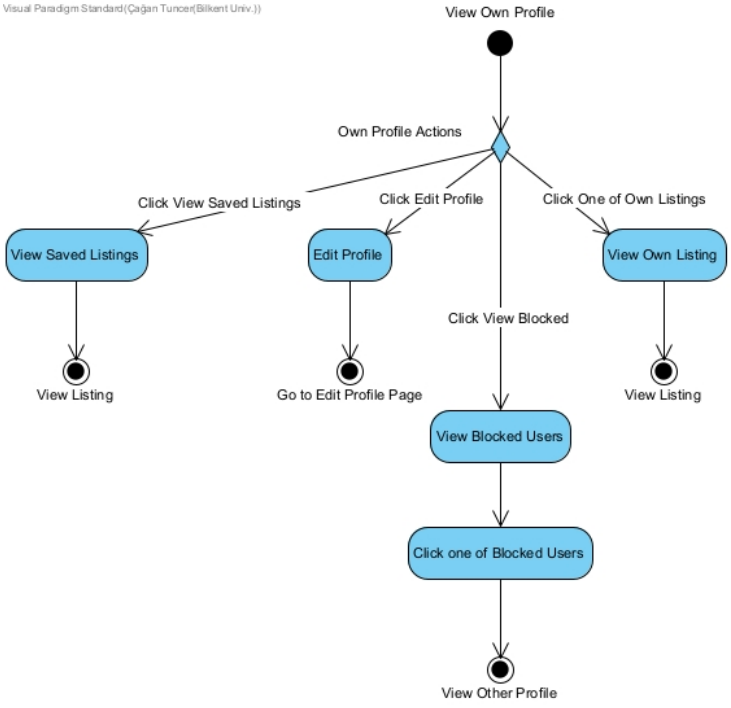
4.4. Listing Activity Diagram



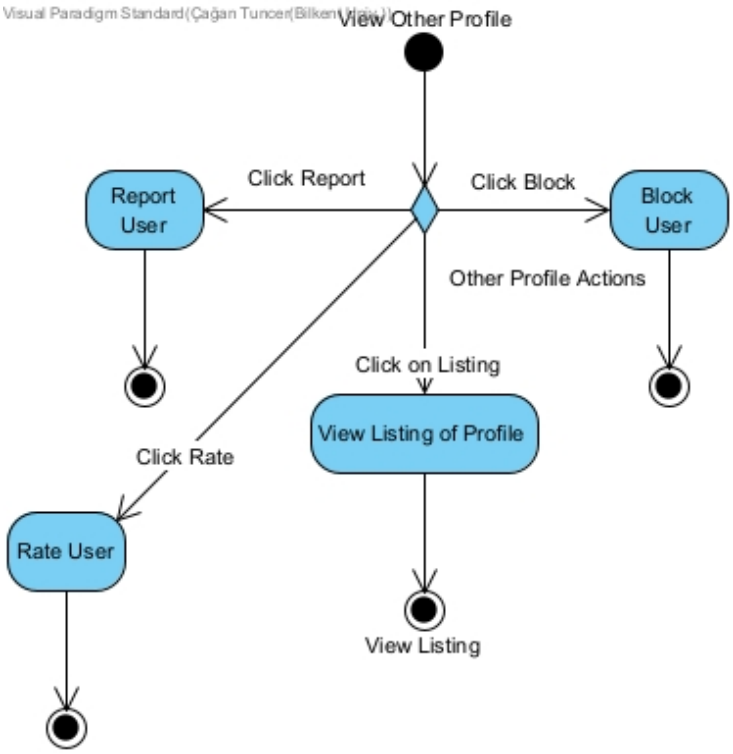
4.5. Edit Profile Activity Diagram



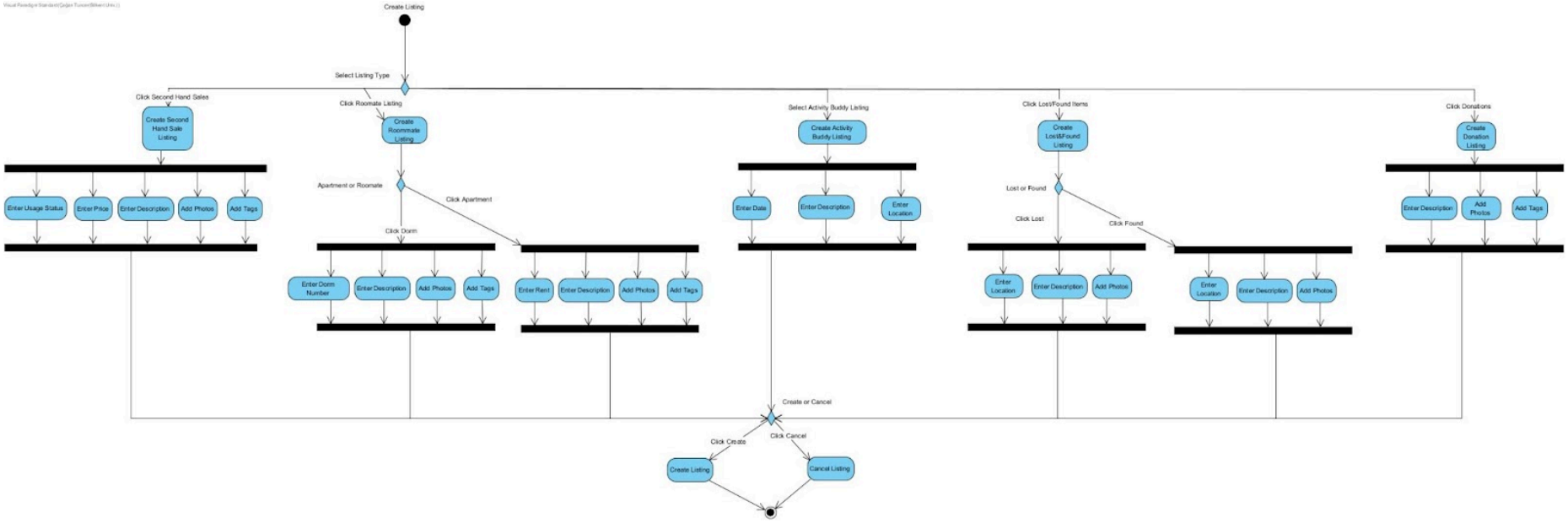
4.6. Own Profile Activity Diagram



4.7. View Other Profile Activity Diagram

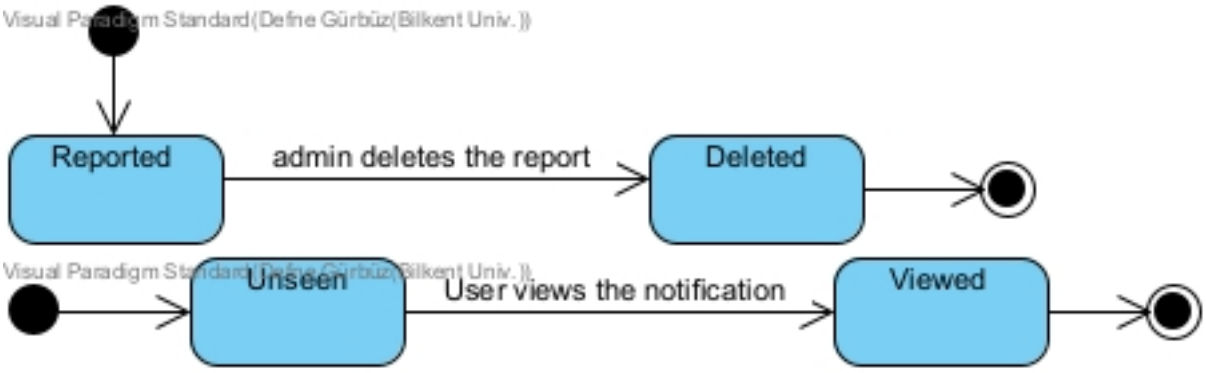


4.8. Create Listing Activity Diagram

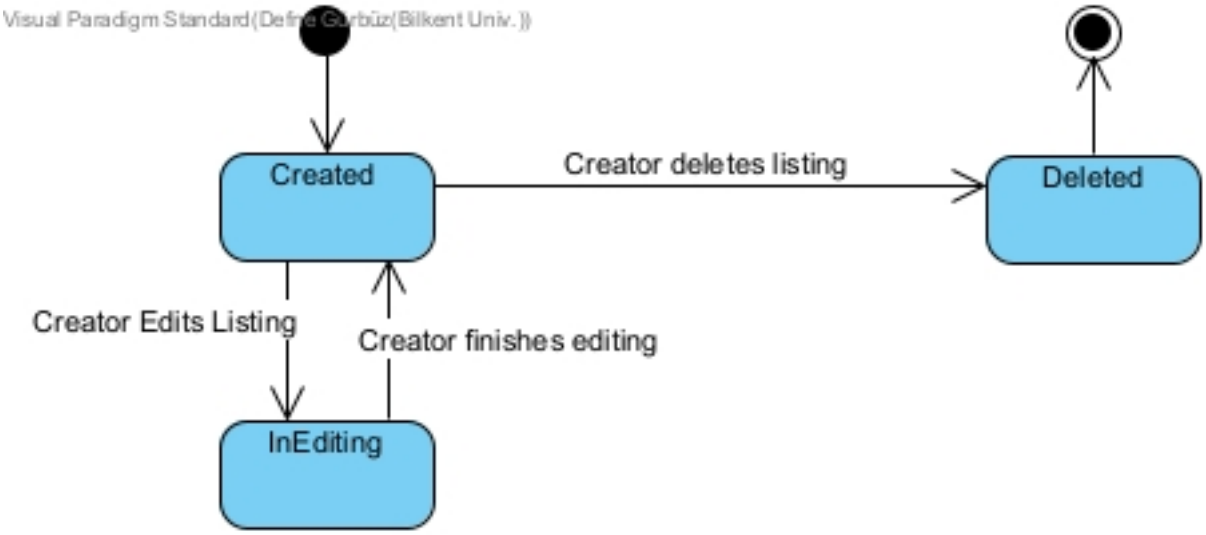


5. State Diagrams

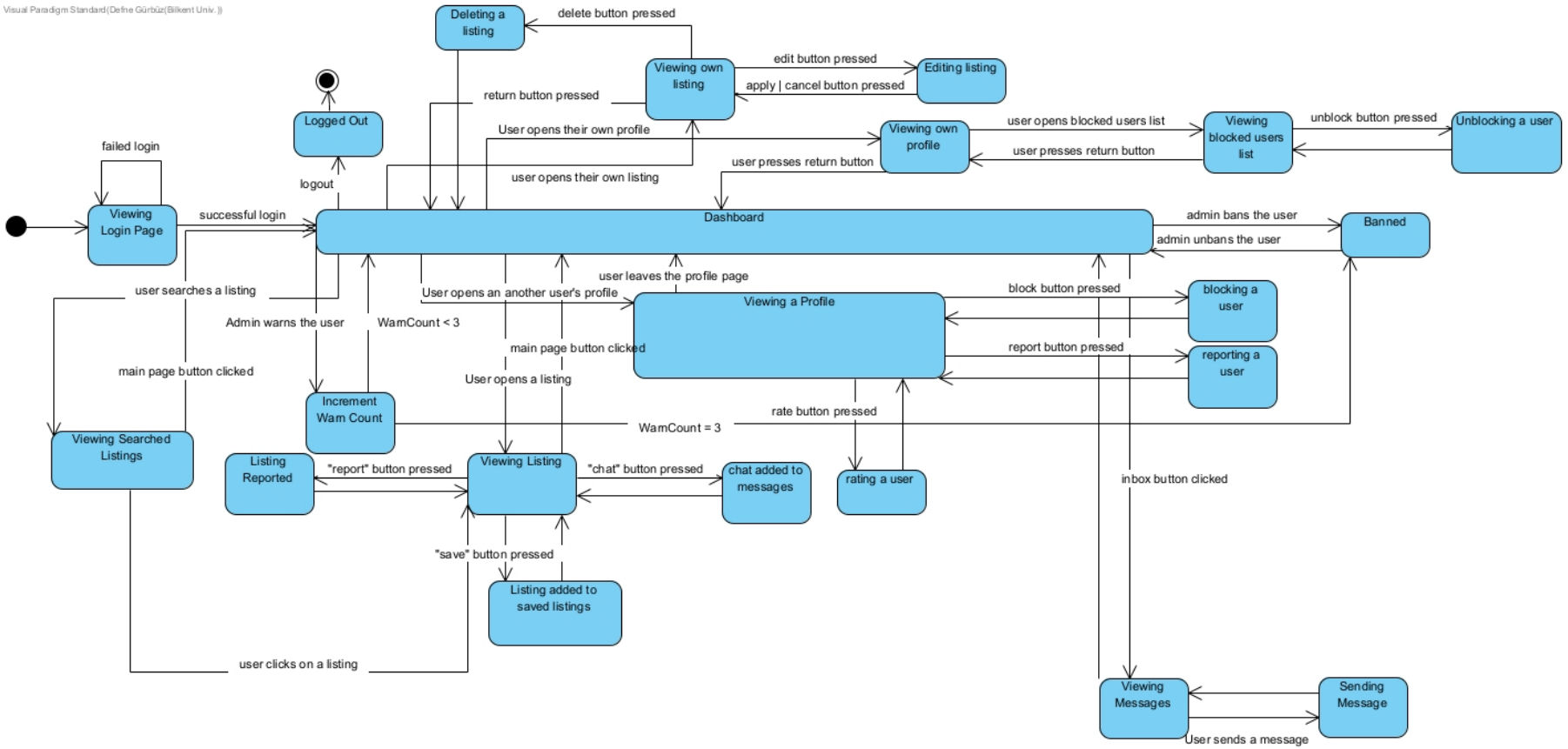
5.1. Report State Diagrams



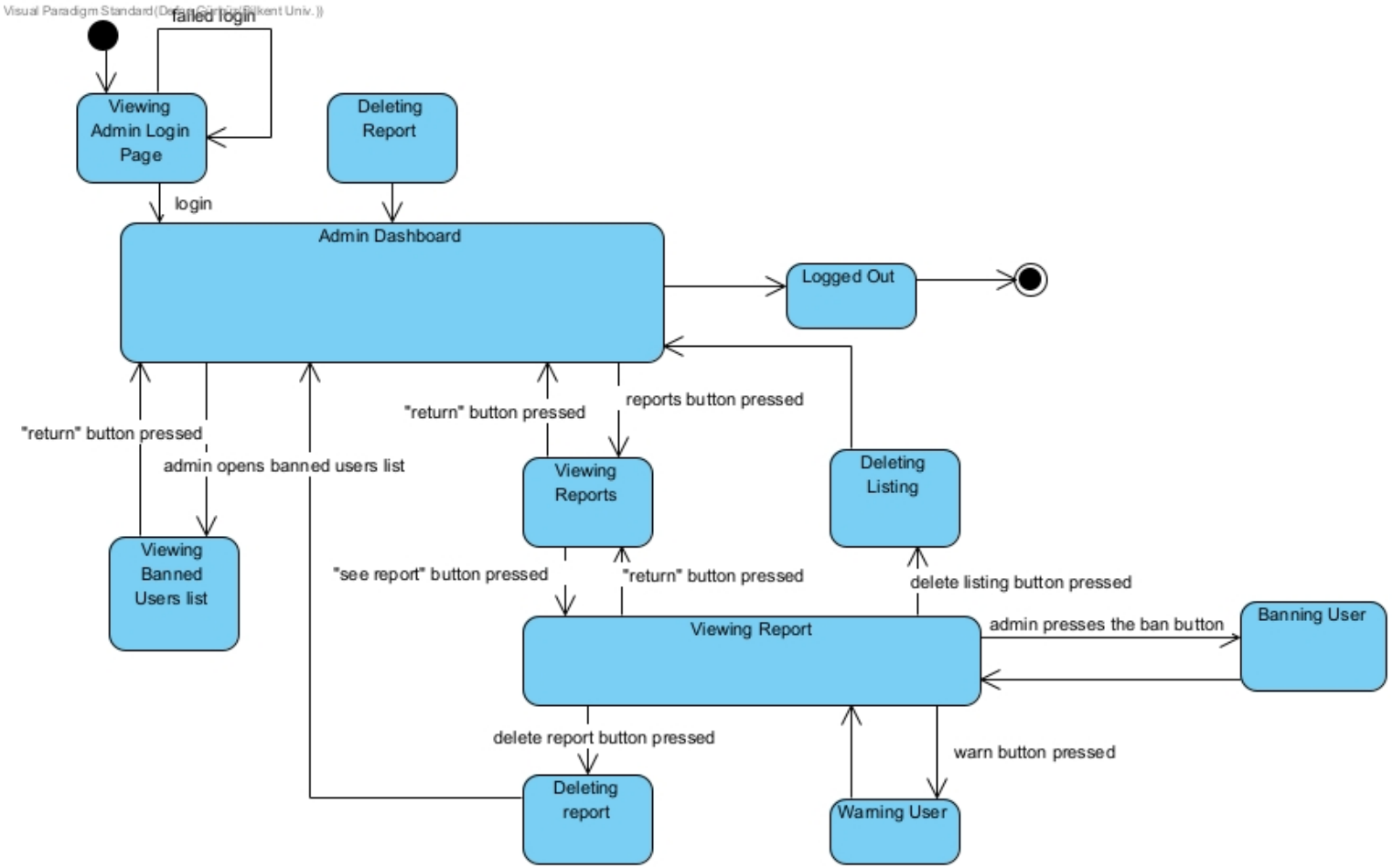
5.2. Listing State Diagrams



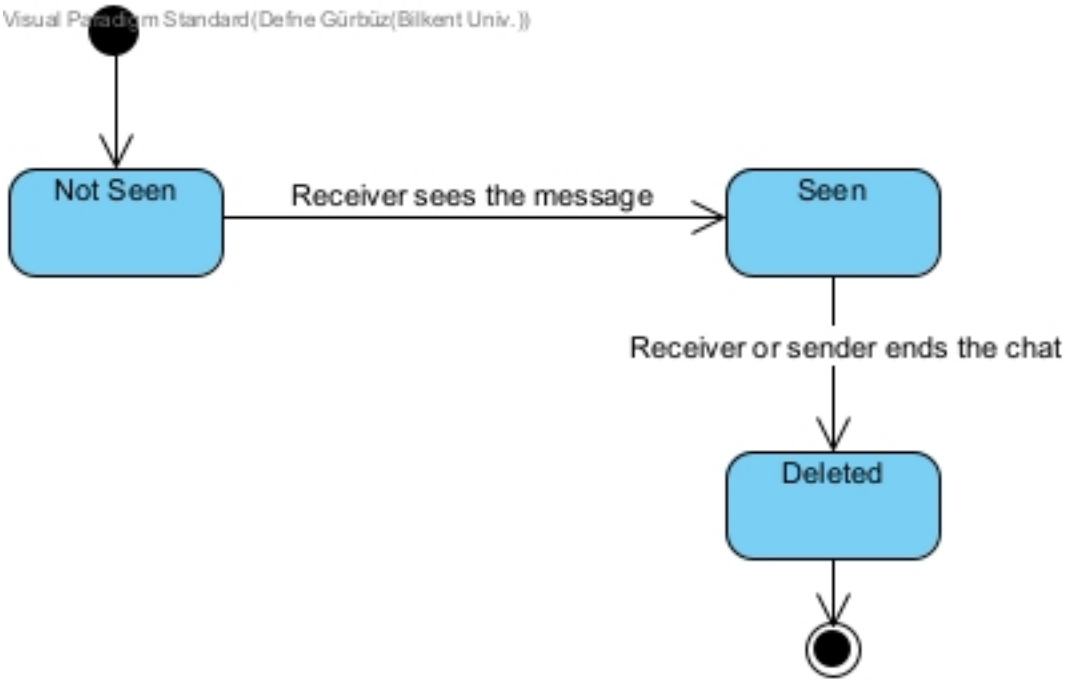
5.3. User State Diagram



5.4. Admin State Diagram



5.5. Message State Diagram



6. Nonfunctional Requirements

- **Usability**
  - When the user performs unintended behavior, the web app should generate understandable error messages.
  - The UI should be easily understandable for anyone that has used a sales website before.
  - The web app should be accessible and properly working from all platforms (PC, iOS, Android etc.).

- **Reliability**
  - The web app should be performing with the same speeds during all times and downtimes should be minimized.

- **Performance**
  - The website should have quick response times to all clicks, under 0.5 seconds.
  - After applying filters or searching for listings, the fitting listings should appear in at most 1.5 seconds.
  - Creating a listing should take at most 1.5 seconds.
  - The website should load in under 2 seconds.

- **Supportability**
  - The web app should receive regular checks and receive updates if necessary.

- **Constraints**
  - The database will be constructed using MongoDB.
  - React will be used for the UI/front end.
  - Node.js and Express.js will be used for the backend.

7. Tech Stack

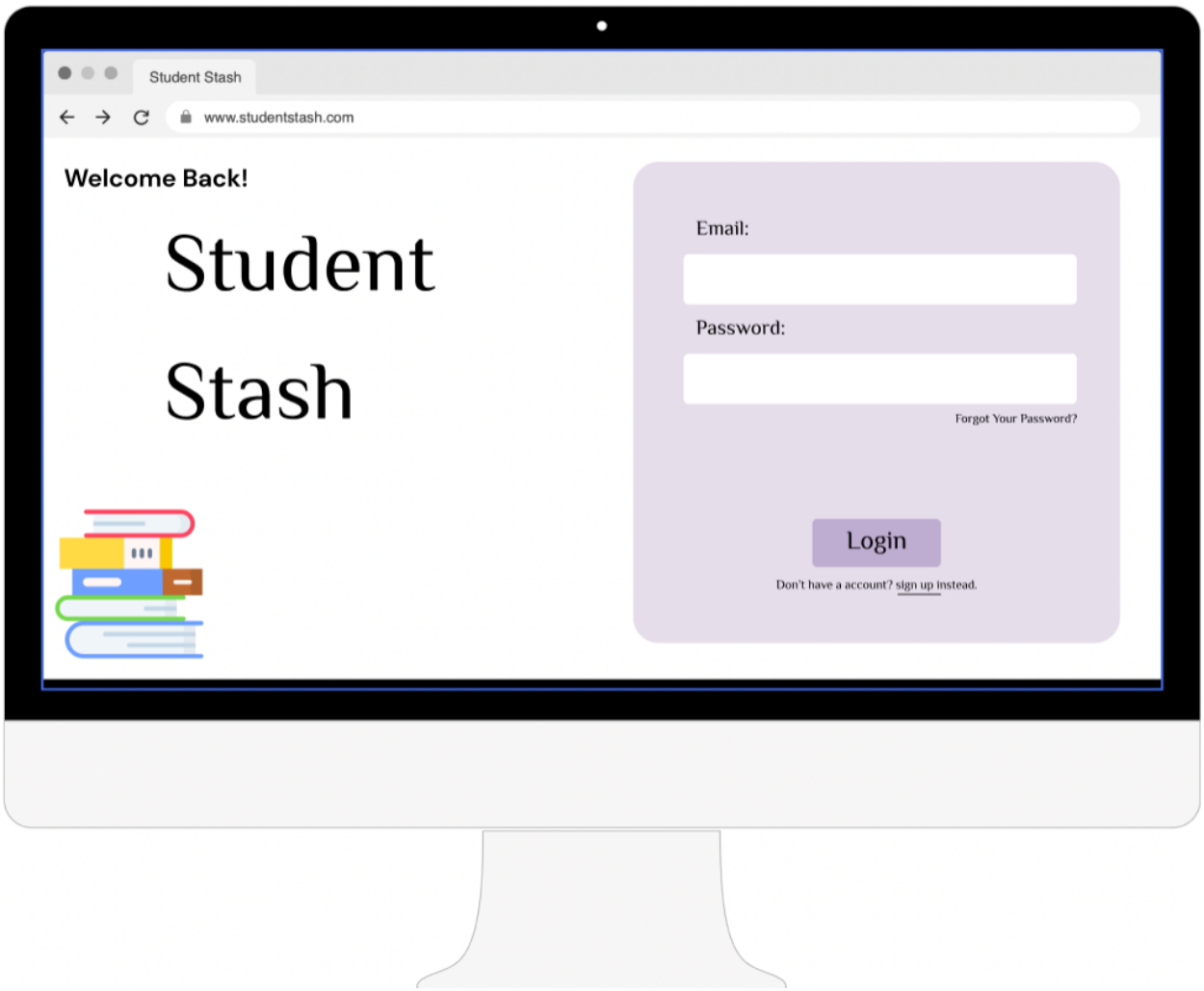
MERN:

- MongoDB for database that will store our web app's data.
- Express(.js) for backend framework.
- React(.js) for frontend framework.
- Node(.js) for the JavaScript web server.

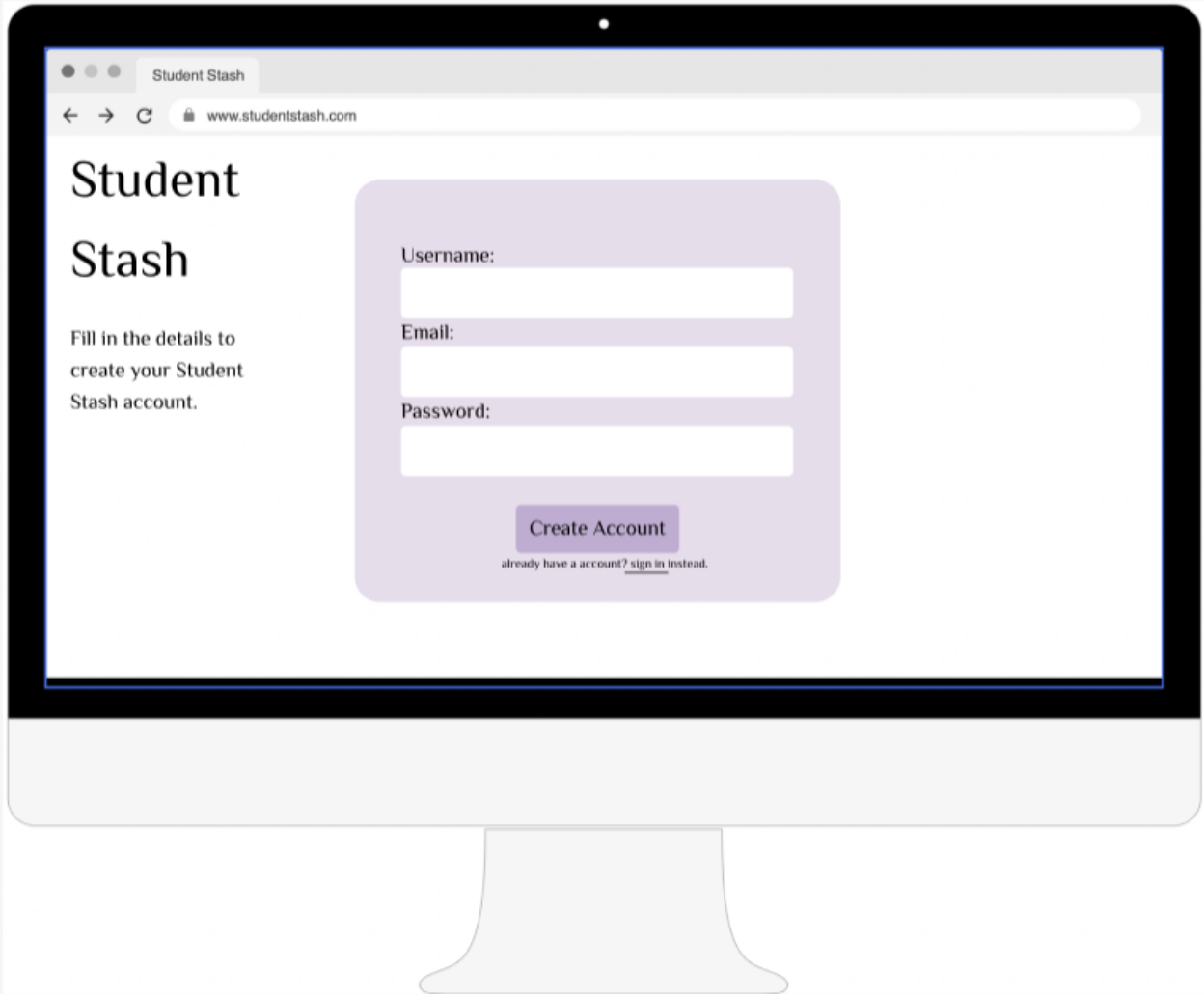
We chose MERN for its usability and efficiency. It will allow us to use JavaScript throughout all of our application stack which will let us reuse our code. It will make our project more flexible and accelerate the development speed

8. Mockups

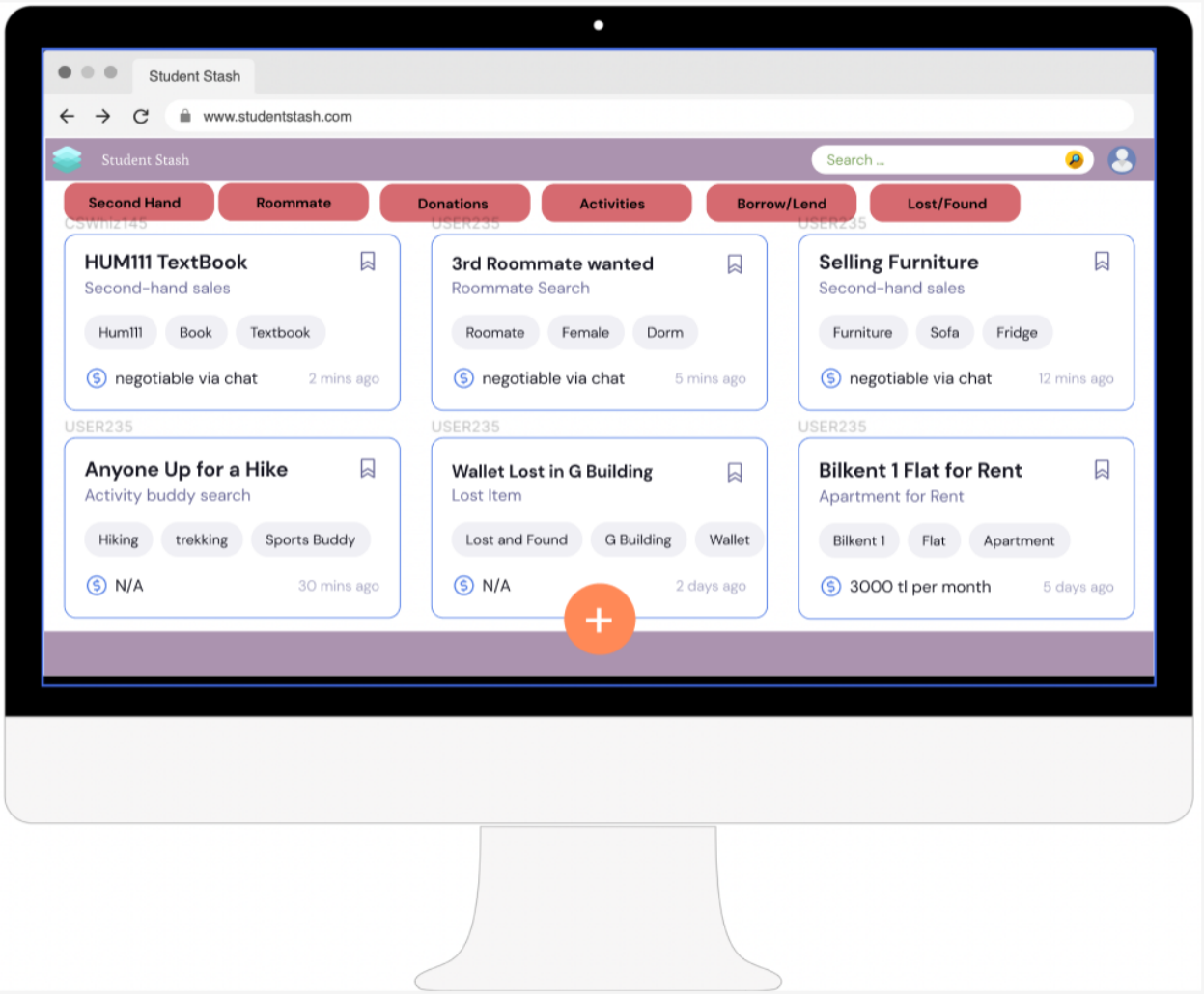
Login Page



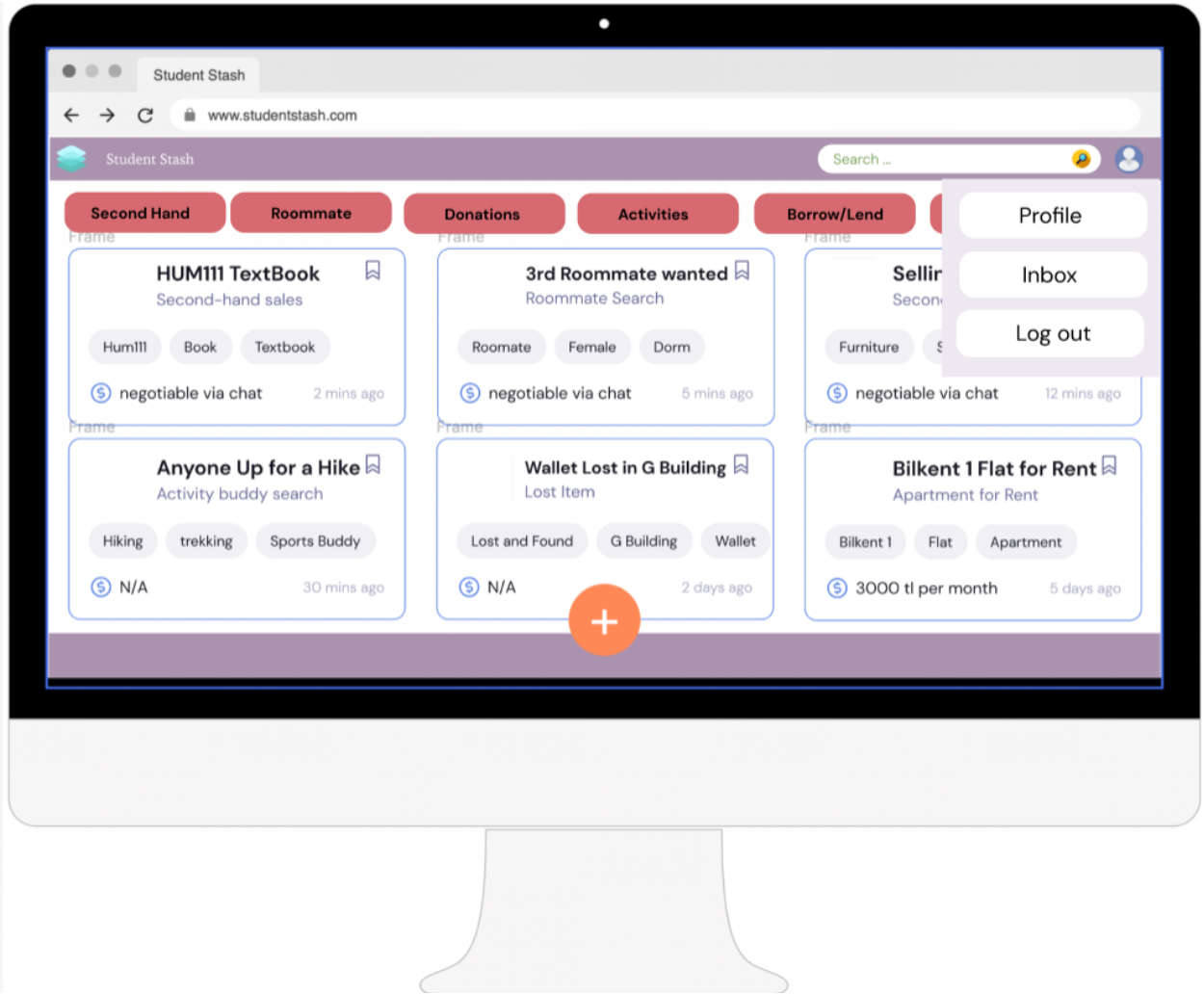
SignUp Page


Student Stash
www.studentstash.com
Student
Stash
Fill in the details to create your Student Stash account.
Username:
Email:
Password:
Create Account
already have a account? sign in instead.

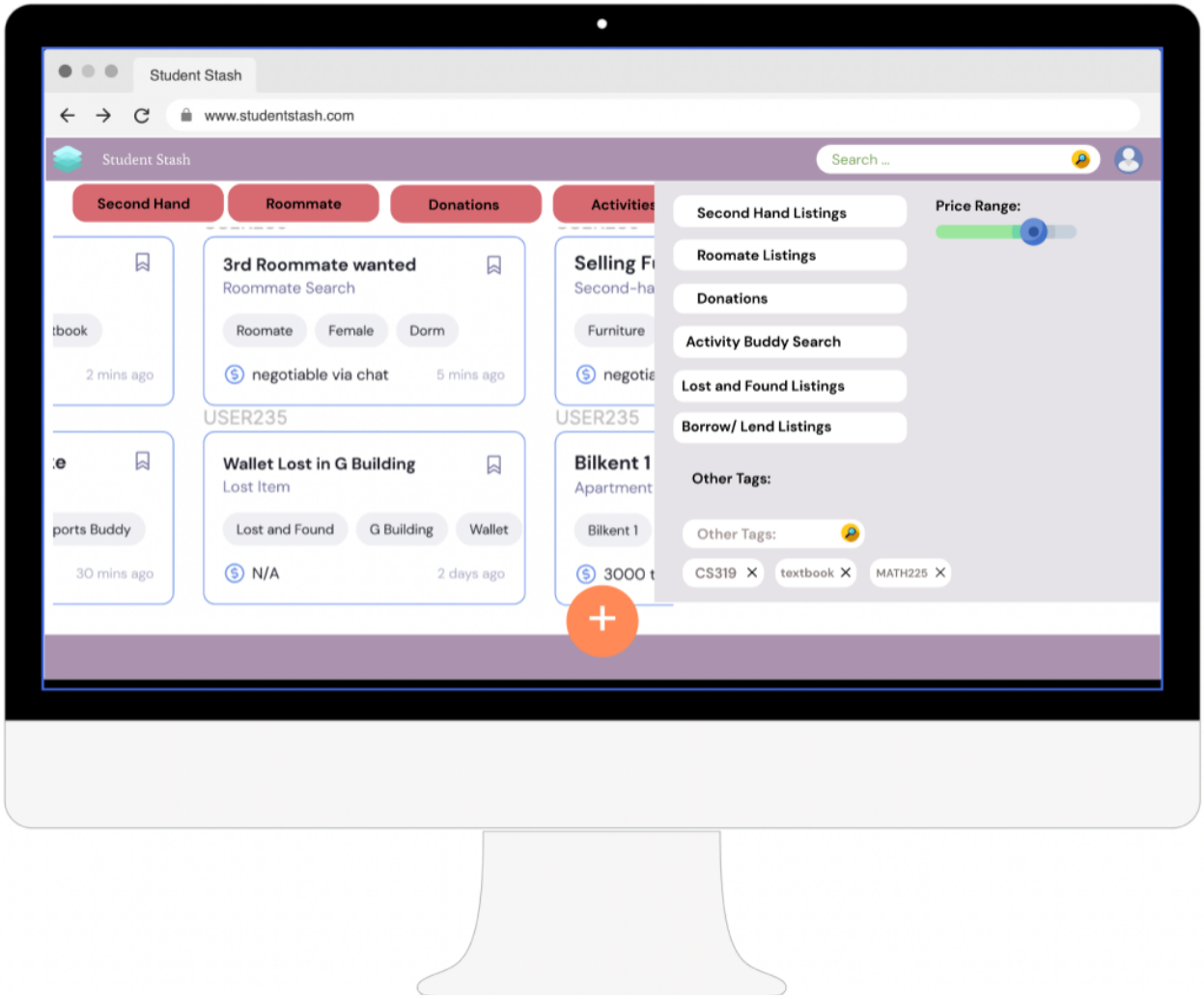
Dashboard


Student Stash
www.studentstash.com
Student Stash
Search ...
Second Hand
Roommate
Donations
Activities
Borrow/Lend
Lost/Found
HUM111 TextBook
Second-hand sales
Hum111
Book
Textbook
negotiable via chat
2 mins ago
3rd Roommate wanted
Roommate Search
Roomate
Female
Dorm
negotiable via chat
5 mins ago
Selling Furniture
Second-hand sales
Furniture
Sofa
Fridge
negotiable via chat
12 mins ago
USER235
Anyone Up for a Hike
Activity buddy search
Hiking
trekking
Sports Buddy
N/A
30 mins ago
USER235
Wallet Lost in G Building
Lost Item
Lost and Found
G Building
Wallet
N/A
2 days ago
USER235
Bilkent 1 Flat for Rent
Apartment for Rent
Bilkent 1
Flat
Apartment
3000 tl per month
5 days ago

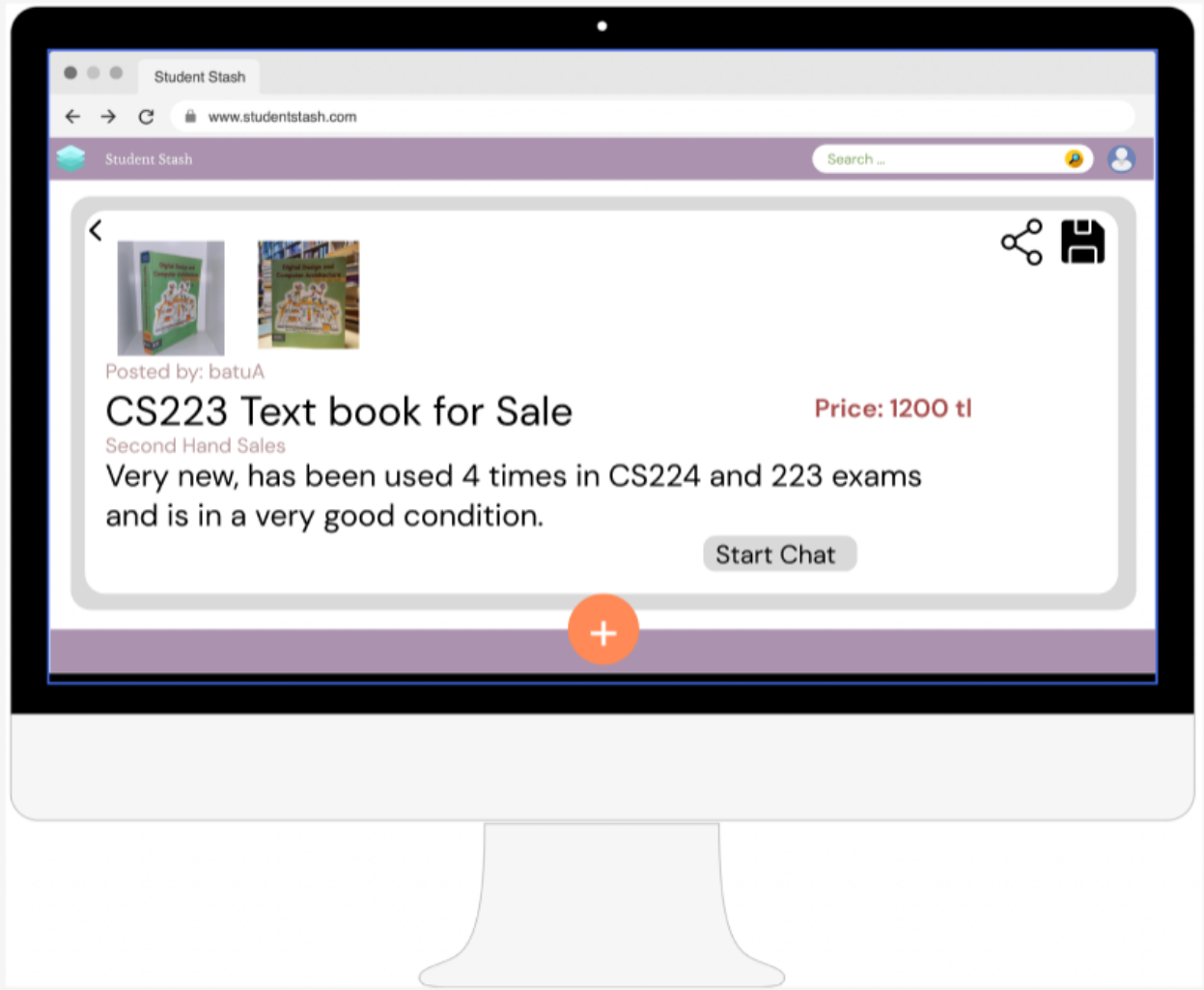
Dashboard


Student Stash
www.studentstash.com
Student Stash
Search ...
Second Hand
Roommate
Donations
Activities
Borrow/Lend
Profile
Inbox
Log out
HUM111 TextBook
Second-hand sales
Hum111
Book
Textbook
negotiable via chat
2 mins ago
3rd Roommate wanted
Roommate Search
Roomate
Female
Dorm
negotiable via chat
5 mins ago
Furniture
negotiable via chat
12 mins ago
Anyone Up for a Hike
Activity buddy search
Hiking
trekking
Sports Buddy
N/A
30 mins ago
Wallet Lost in G Building
Lost Item
Lost and Found
G Building
Wallet
N/A
2 days ago
Bilkent 1 Flat for Rent
Apartment for Rent
Bilkent 1
Flat
Apartment
3000 tl per month
5 days ago

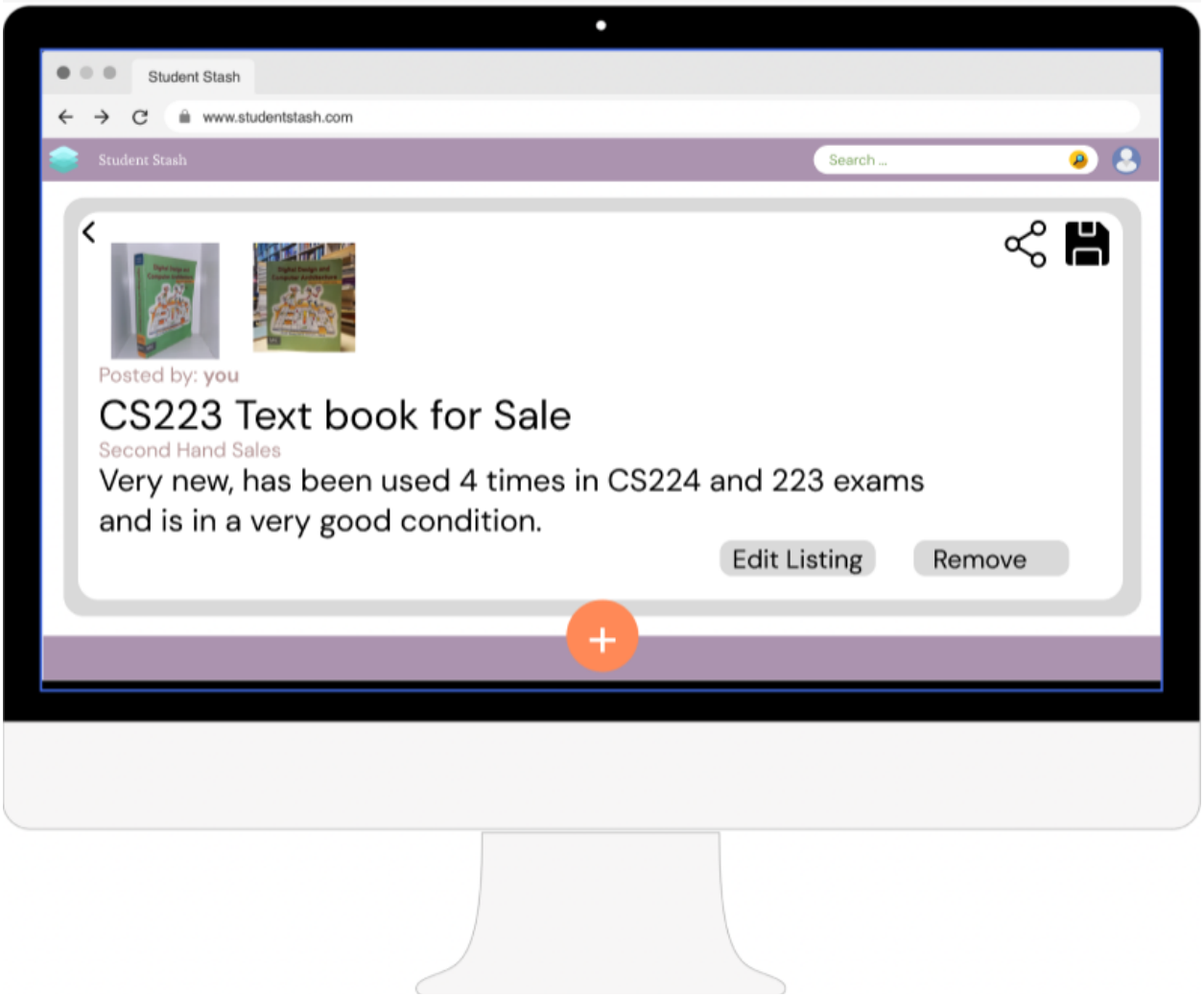
Filters/Search Bar



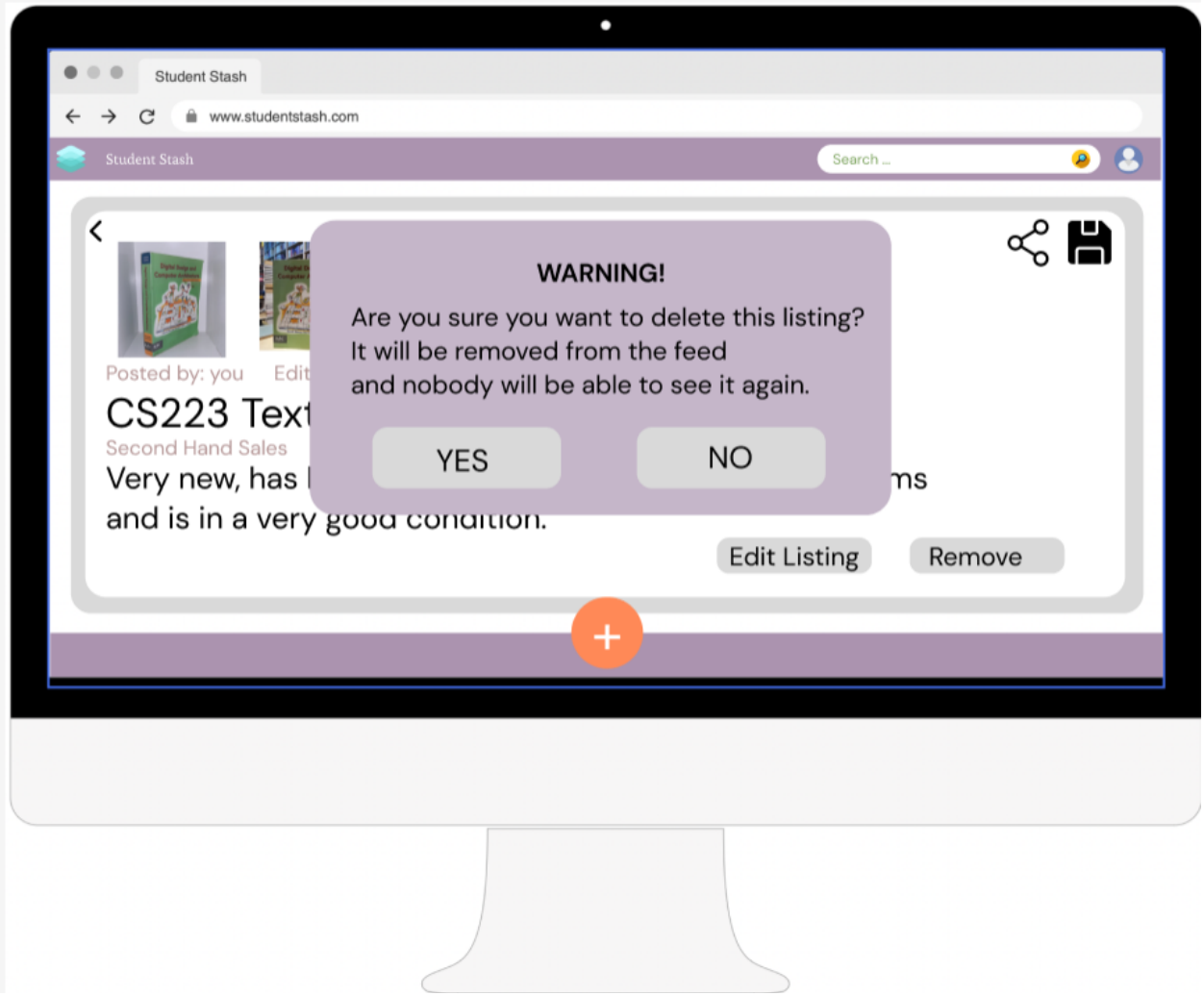
Other User's Listing Page



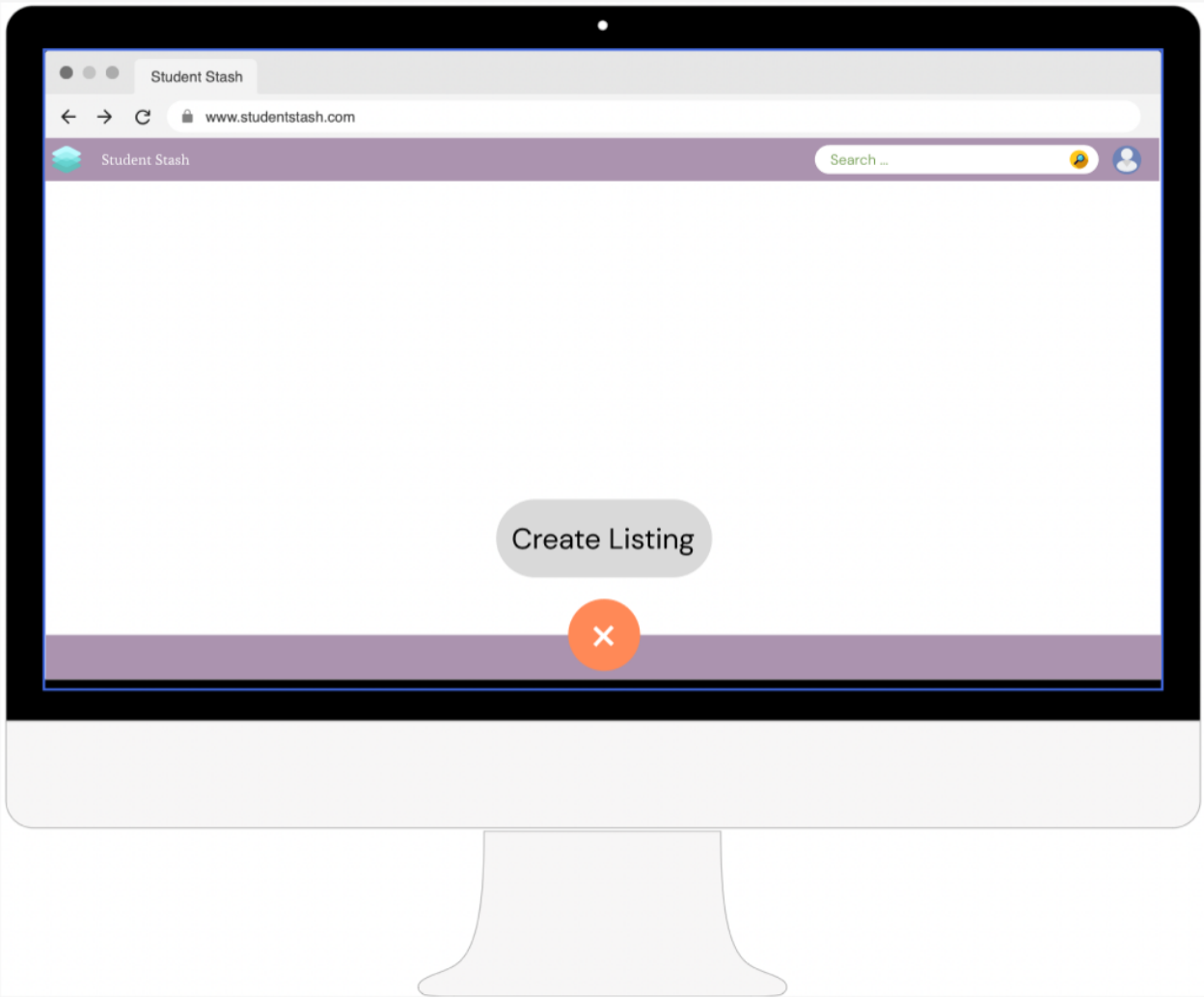
Your Listing Page



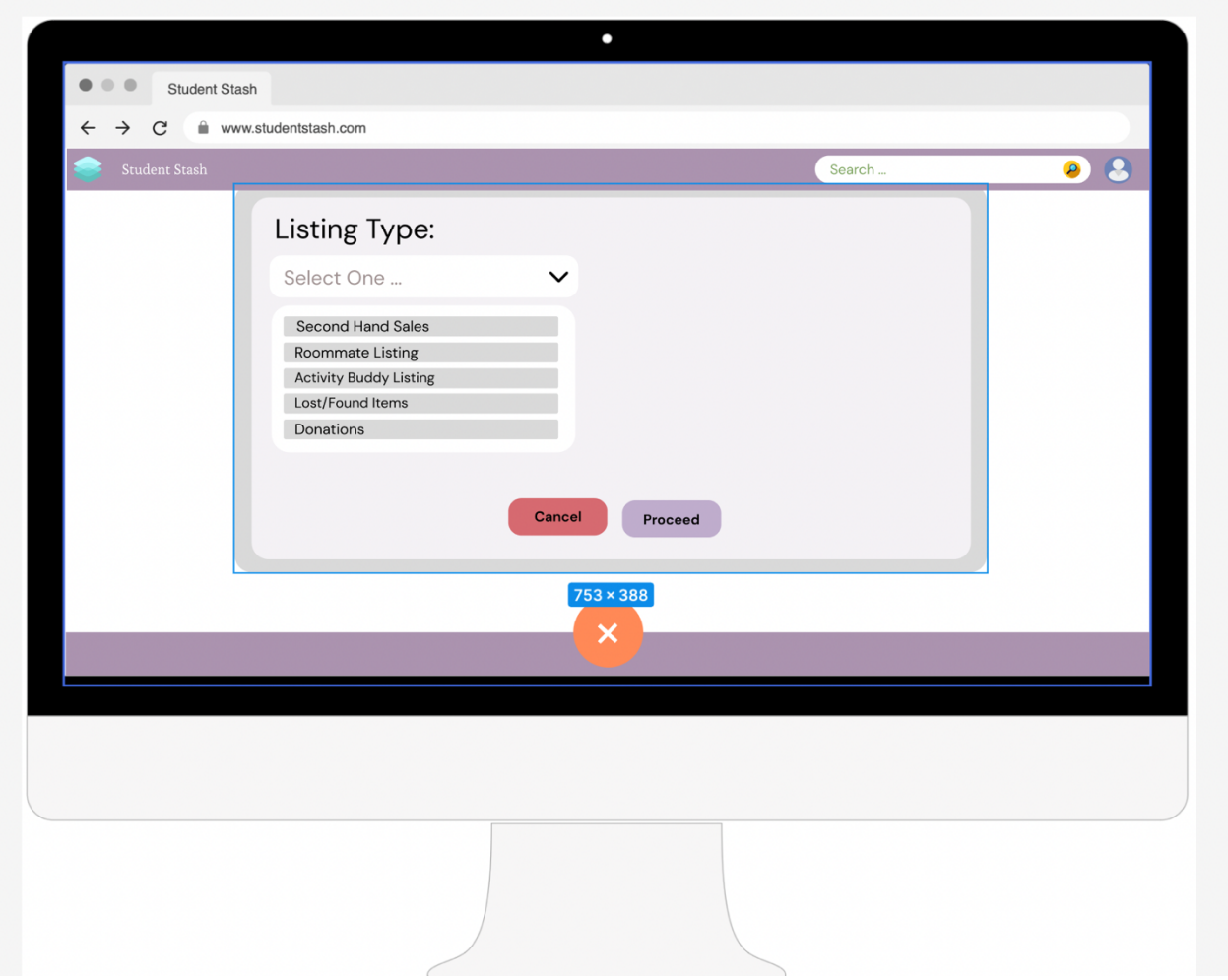
Deleting Your Listing



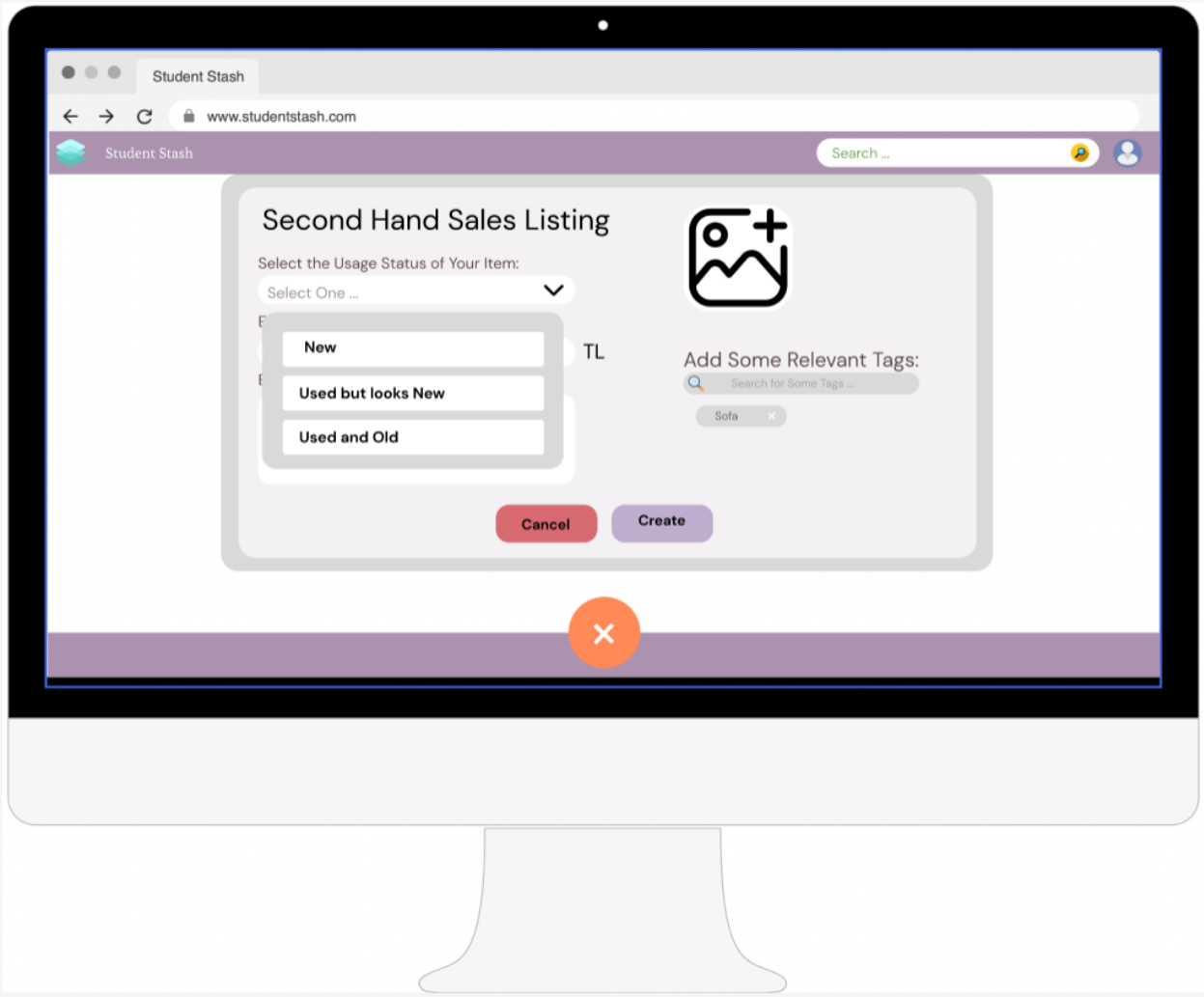
Create Listing



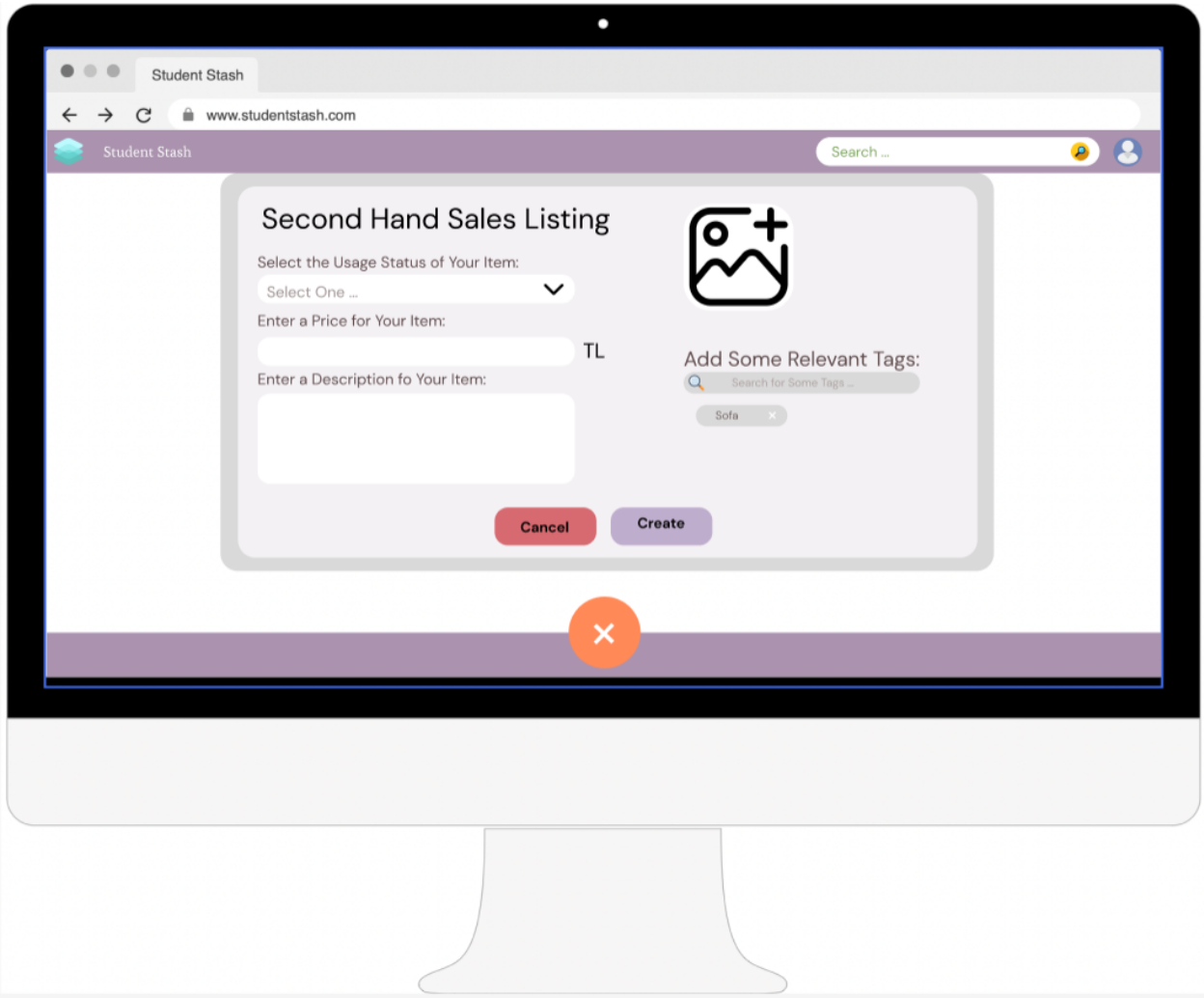
Listing Page



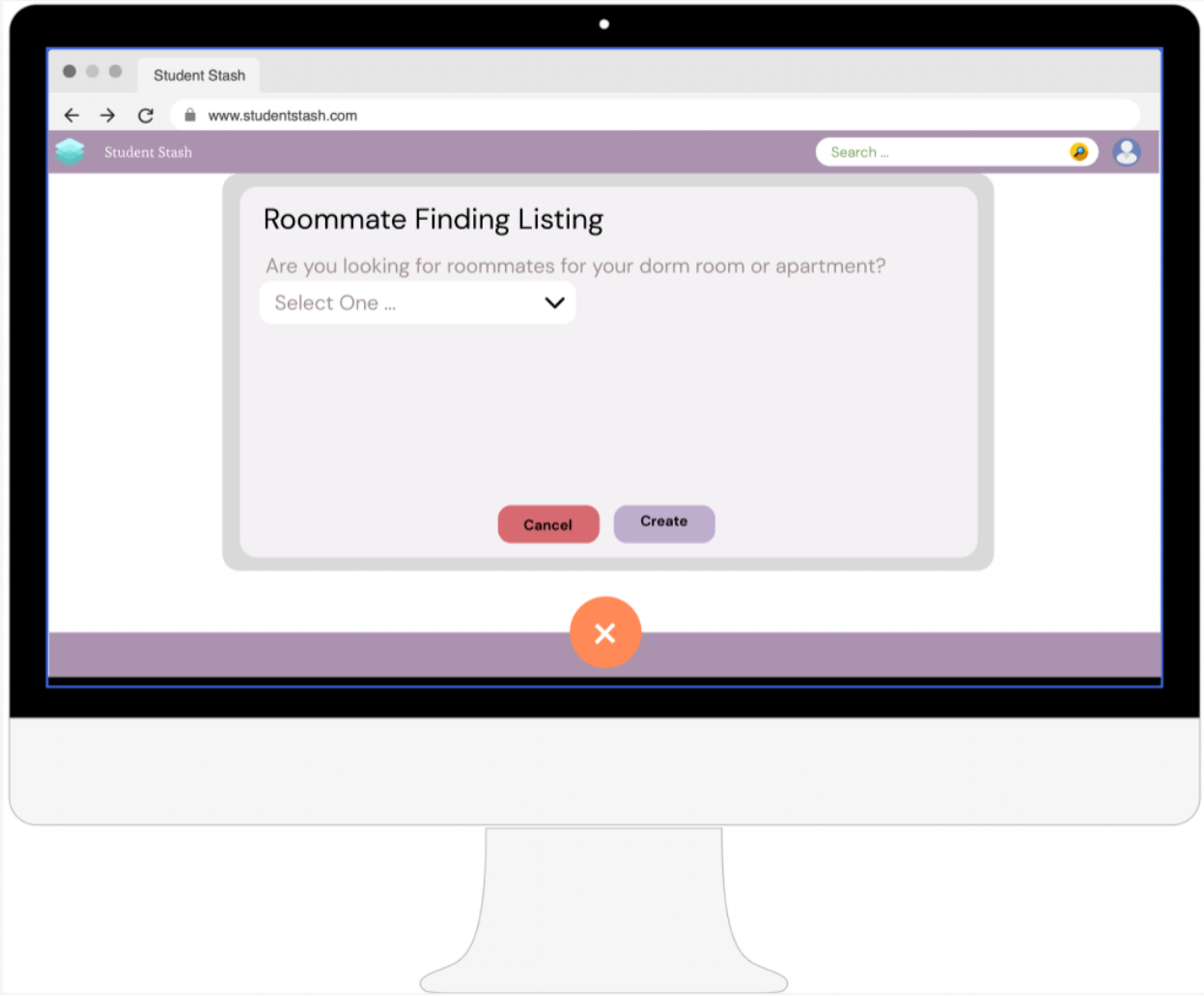
Second Hand Listing Page



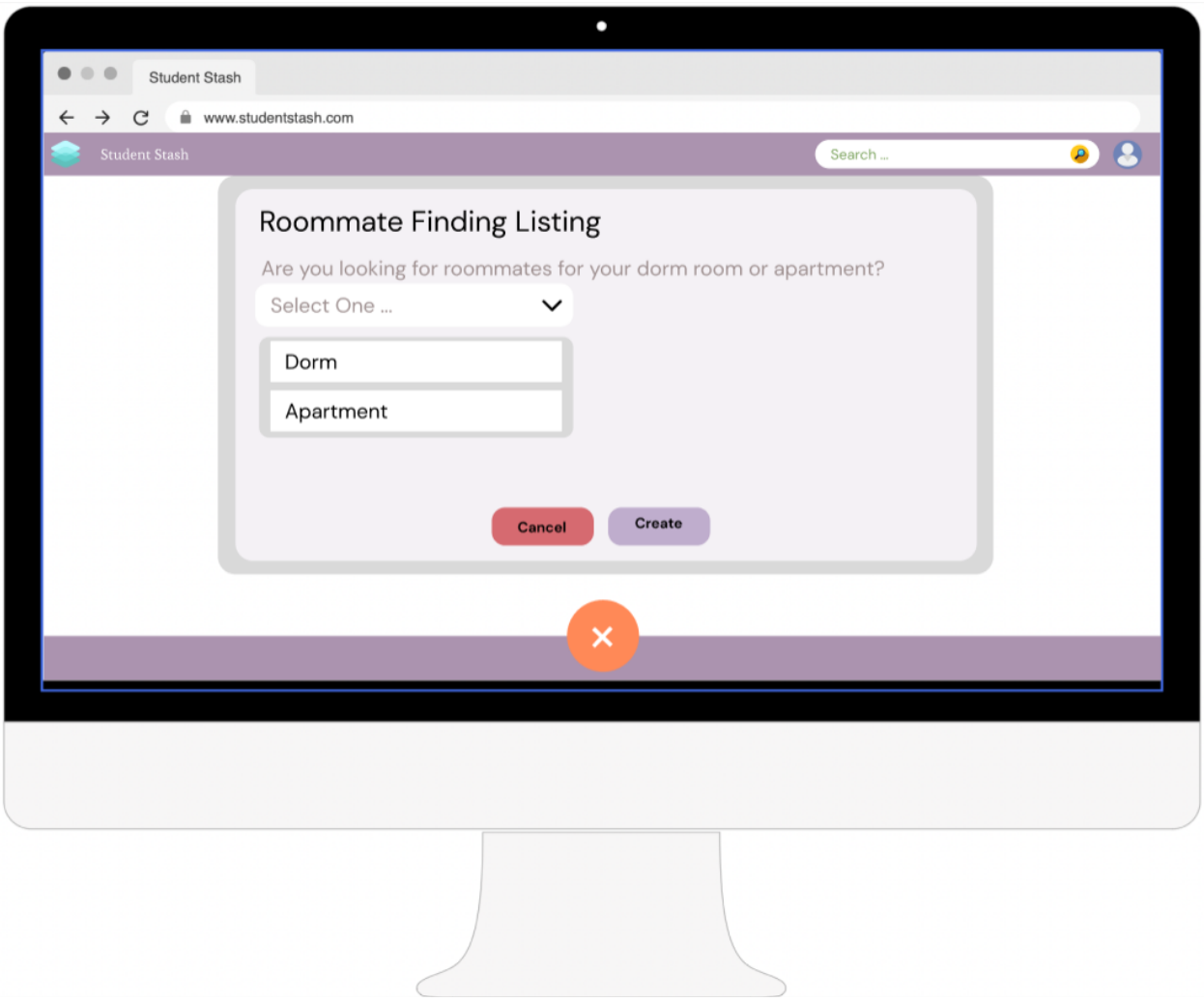
Second Hand Listing Page



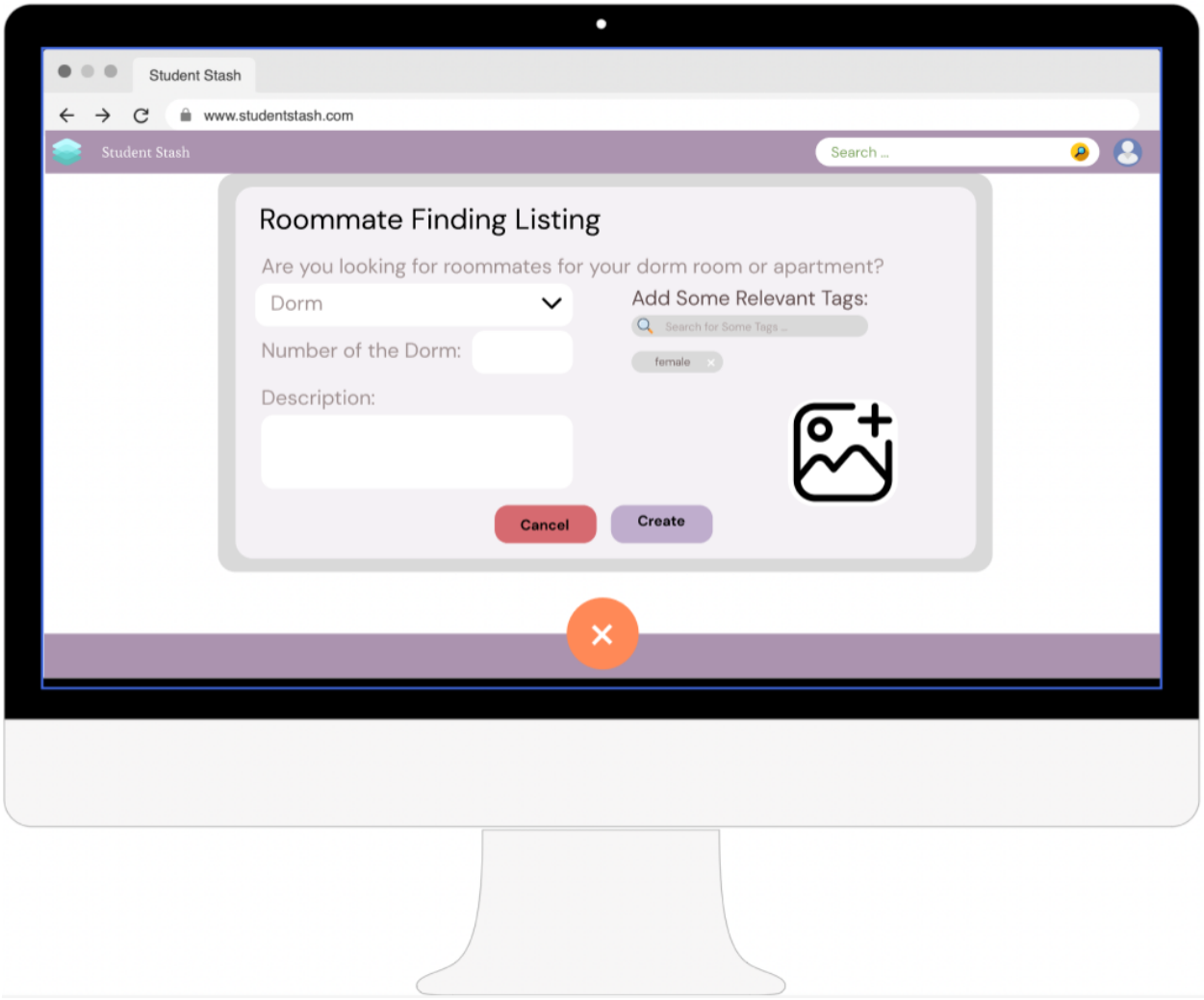
Roommate Finding Listing Page



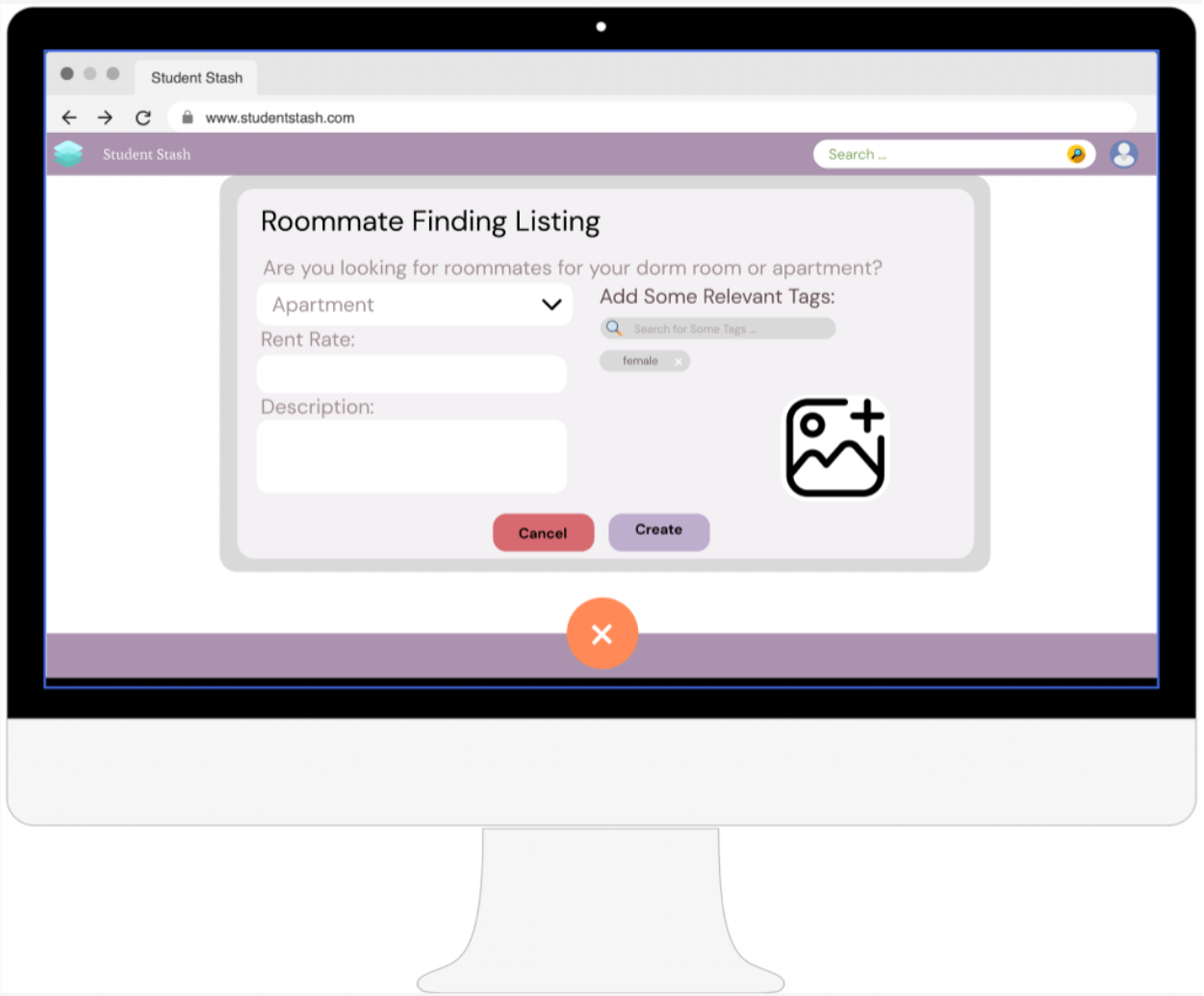
Roommate Finding Listing Page



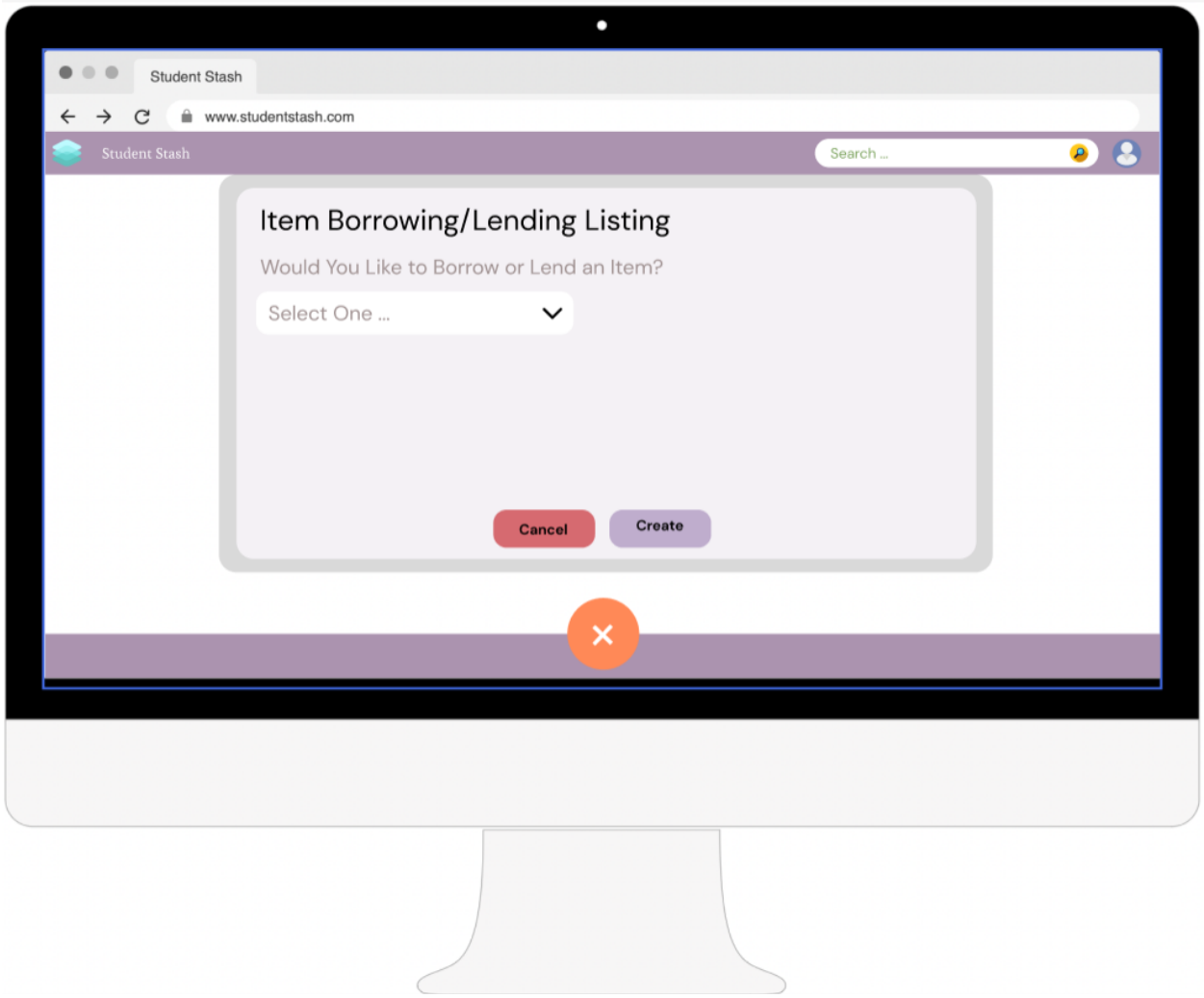
Roommate Finding Listing Page



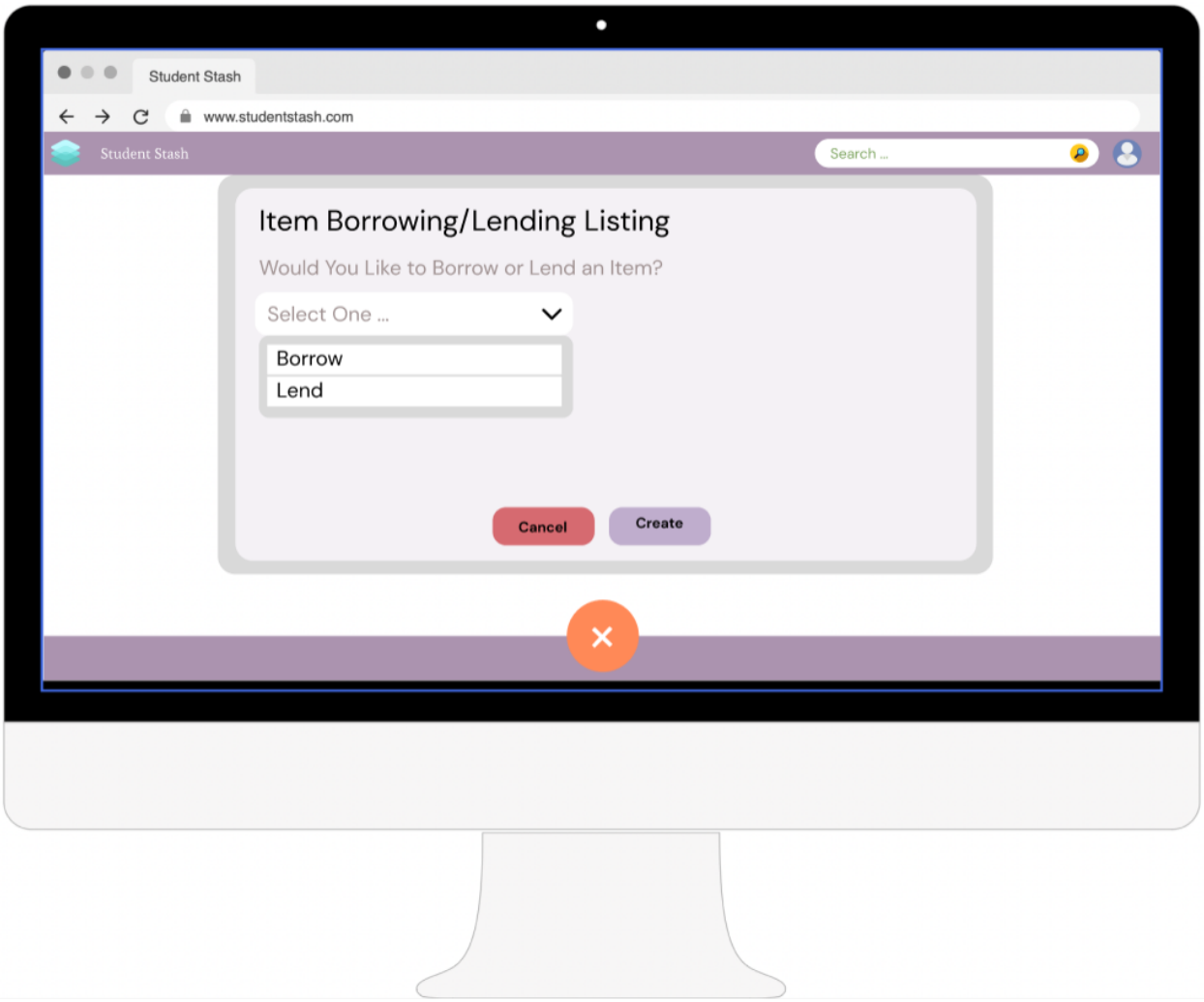
Roommate Finding Listing Page



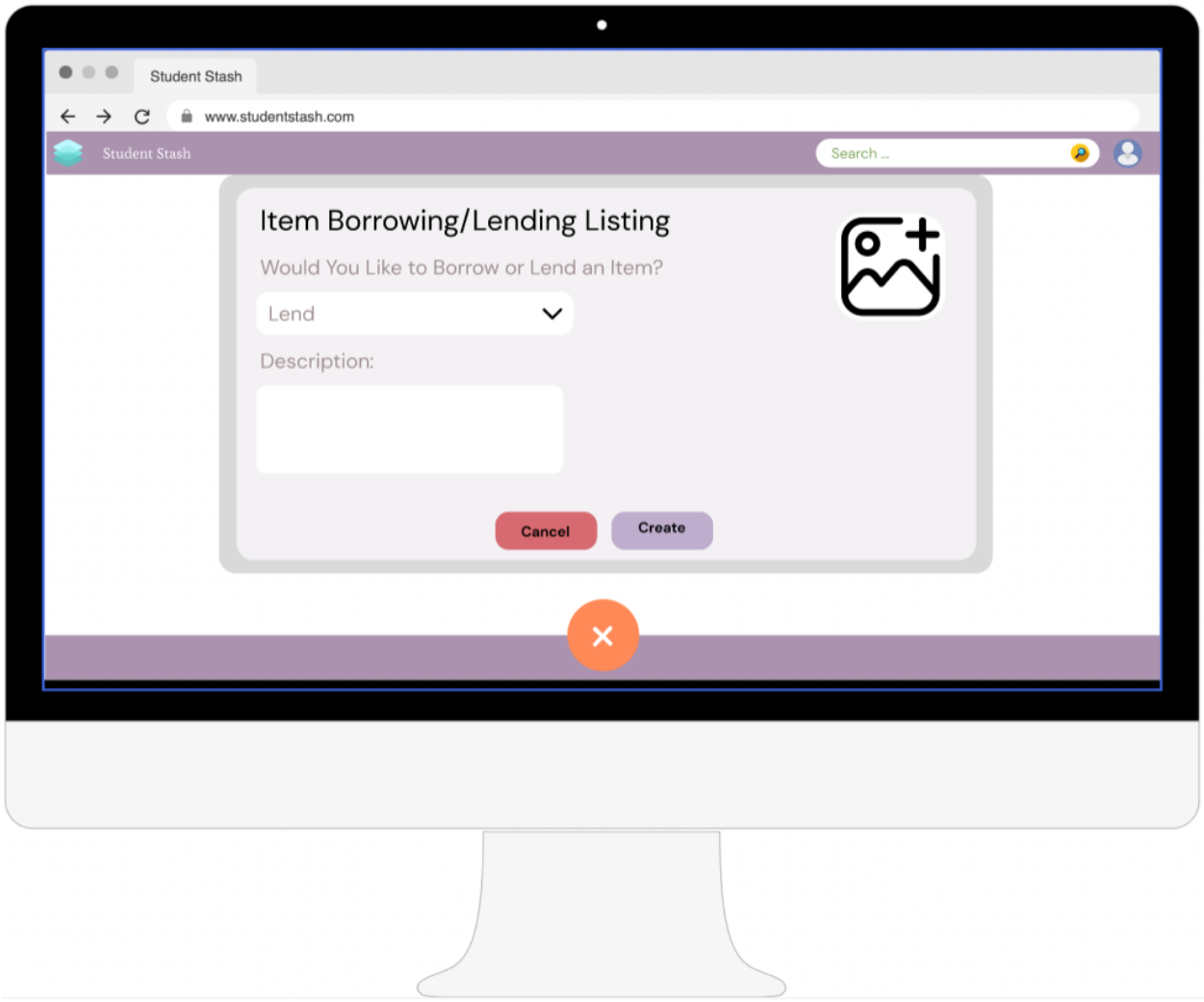
Borrowing/Lending Listing



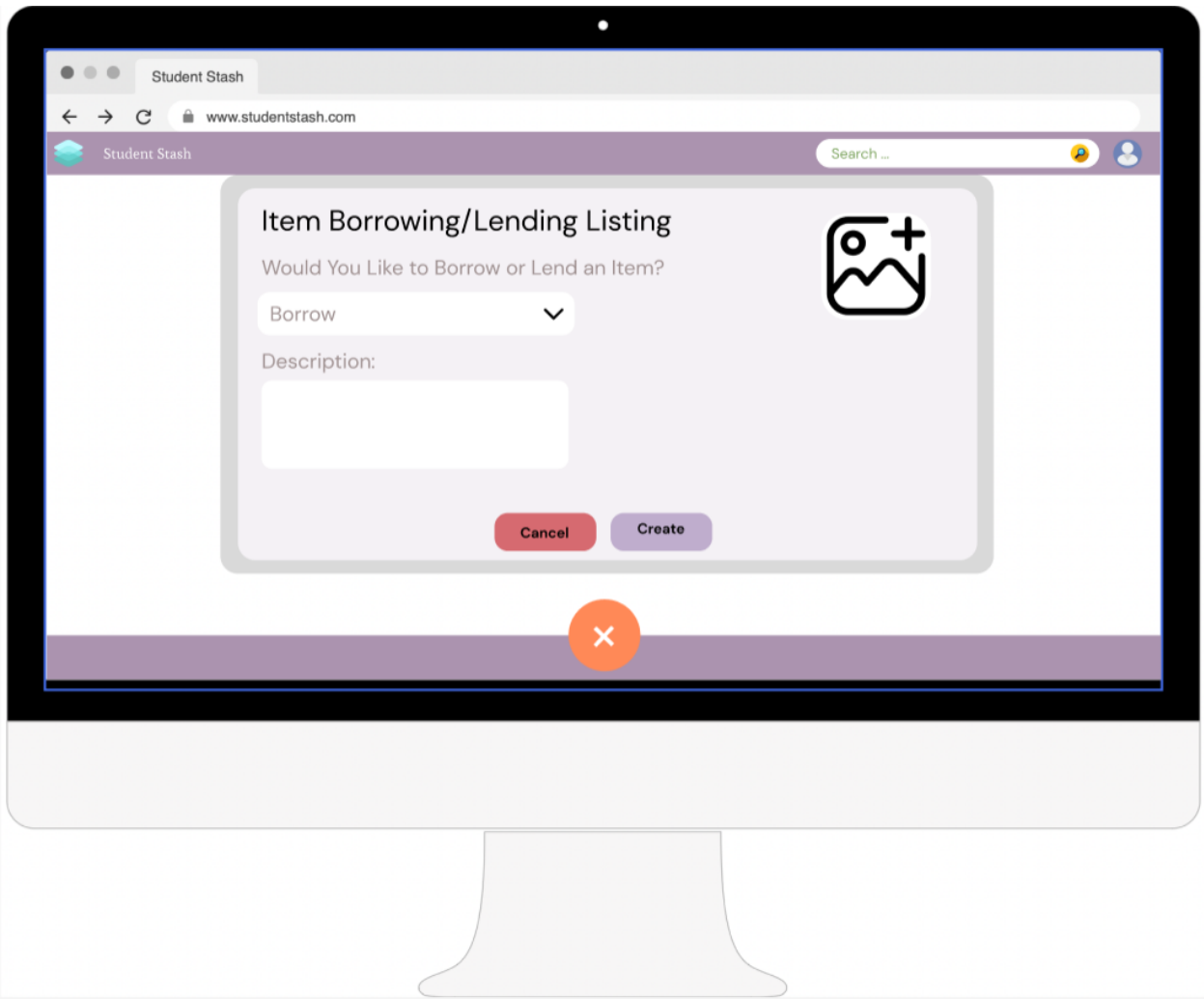
Borrowing/Lending Listing



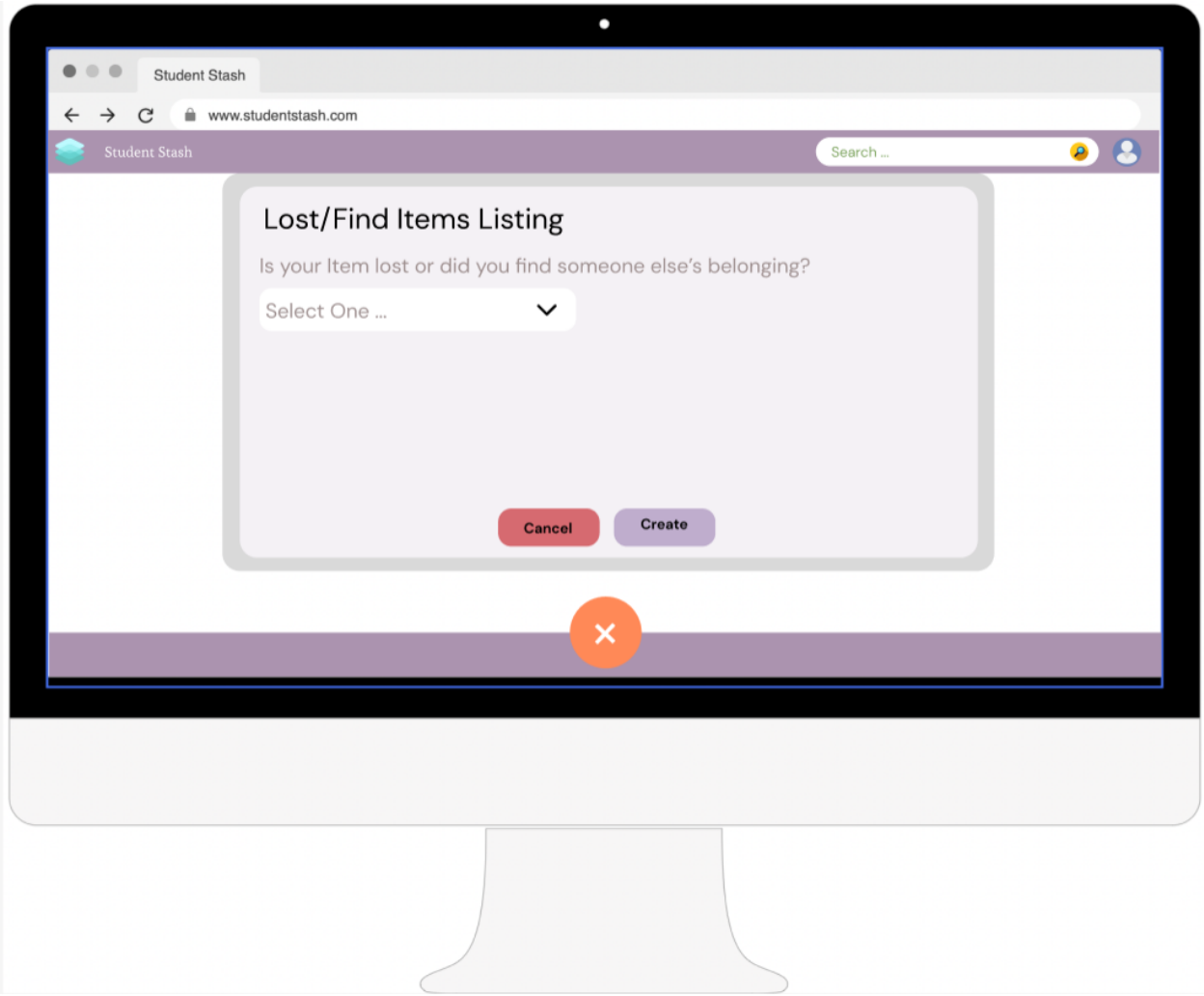
Lending Listing



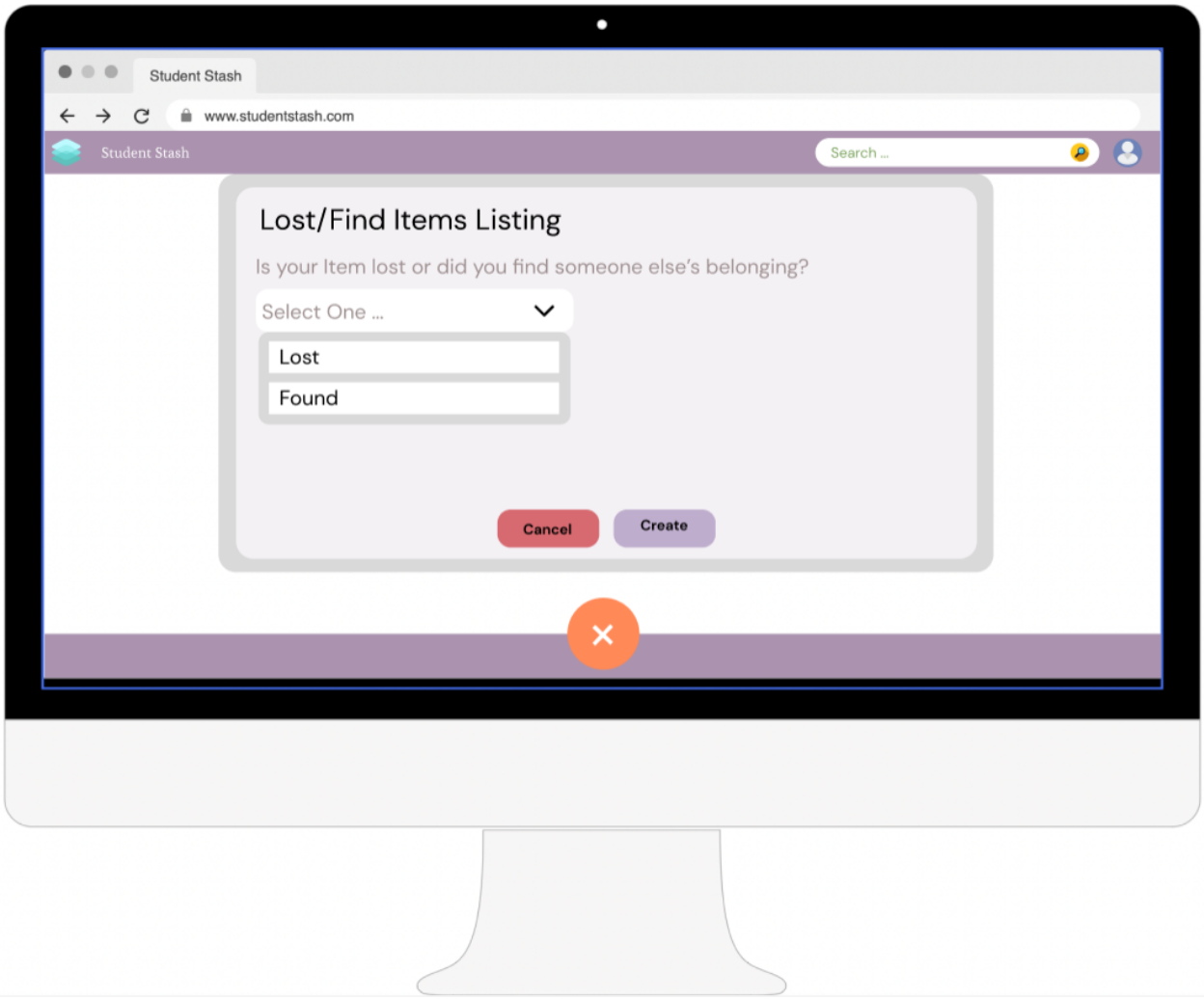
Borrowing Listing



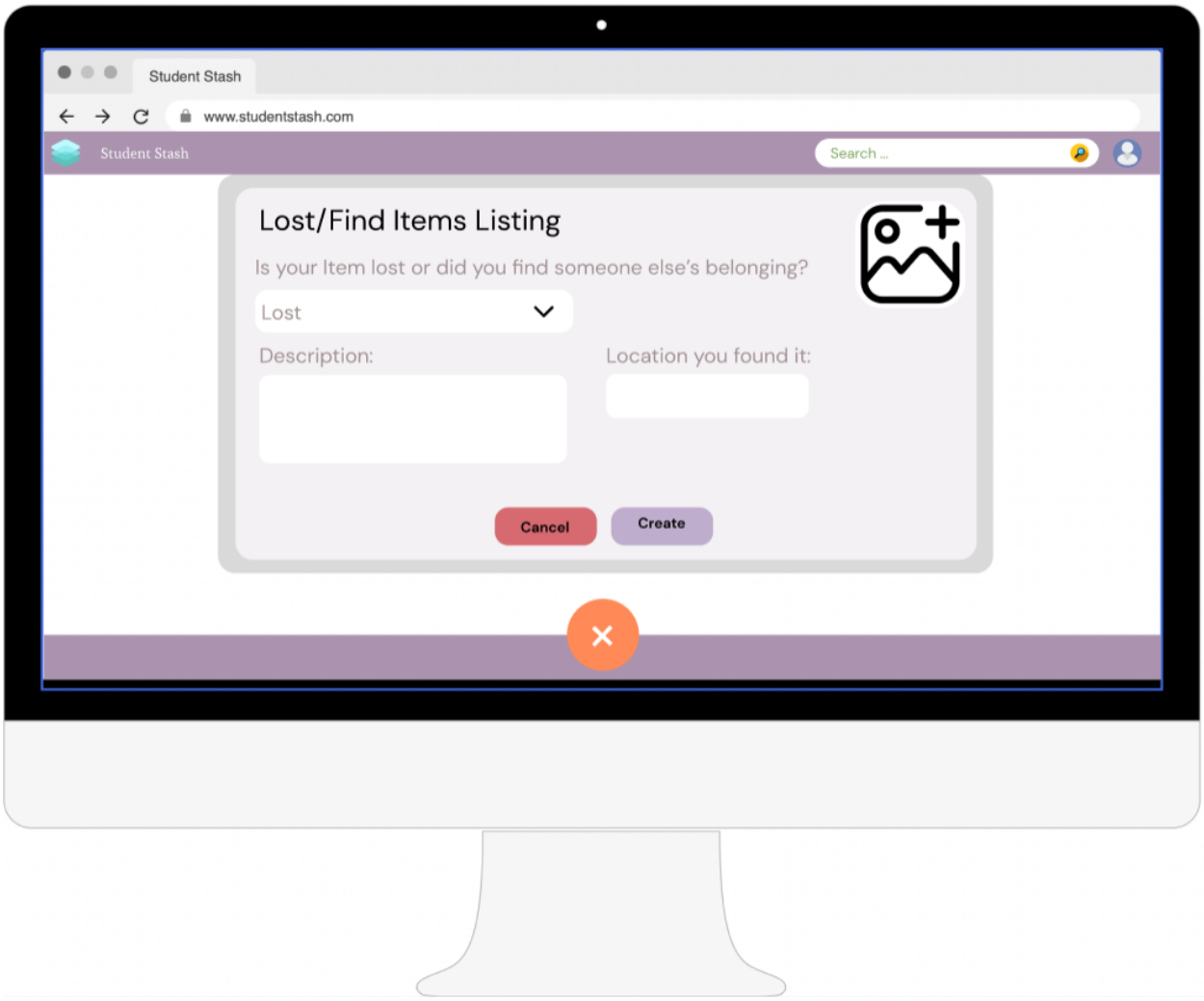
Lost/Found Listing



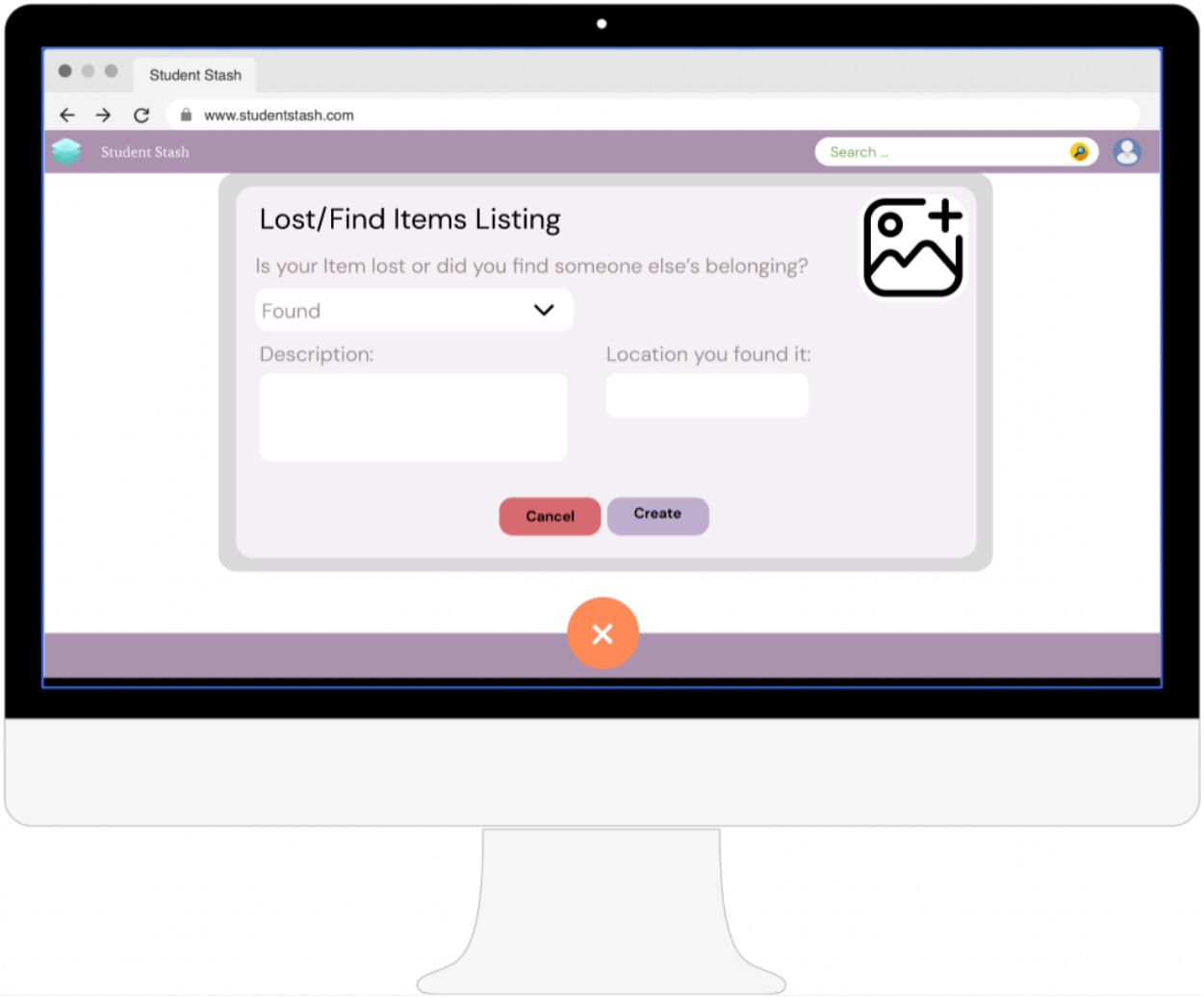
Lost/Found Listing



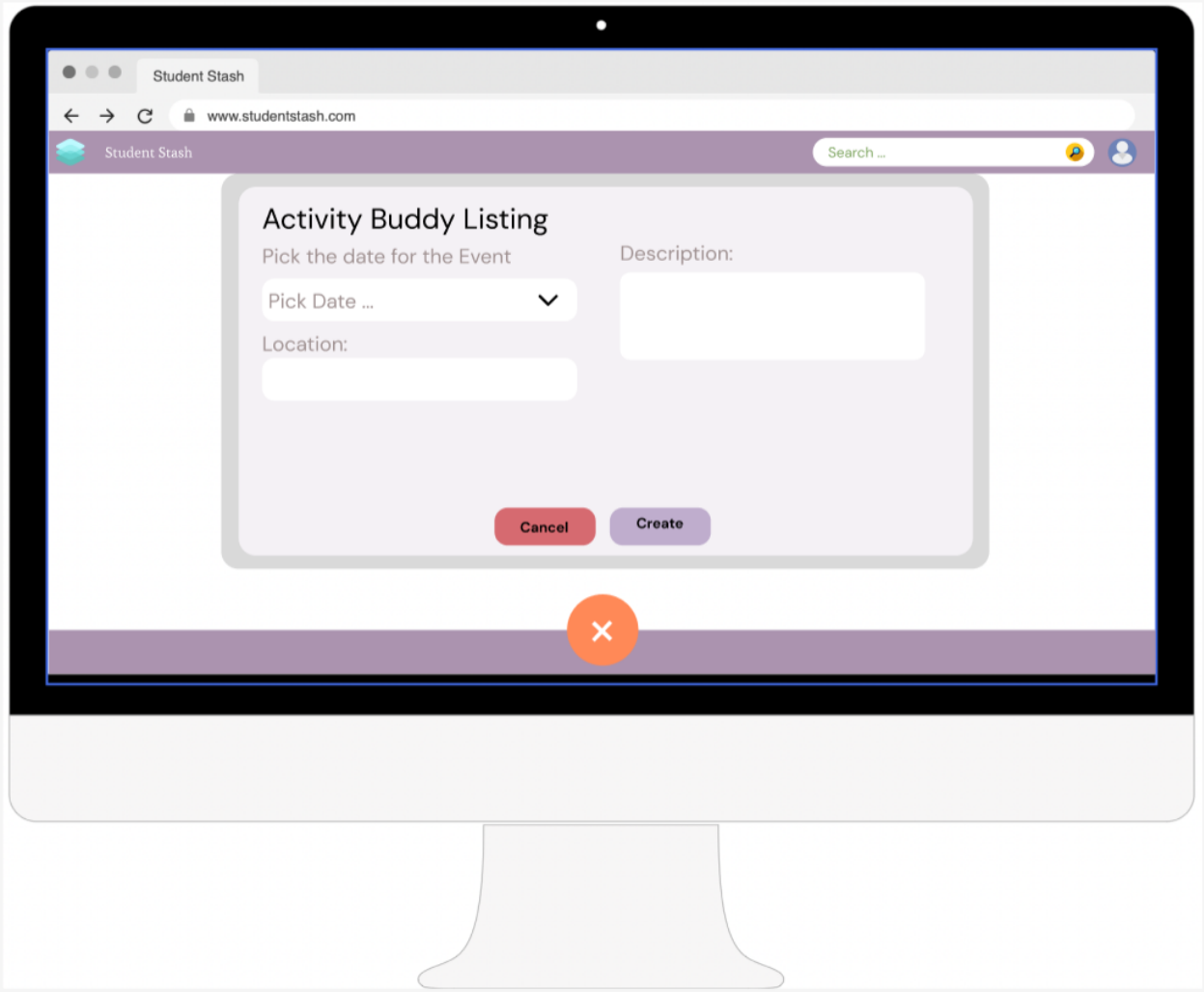
Lost Listing



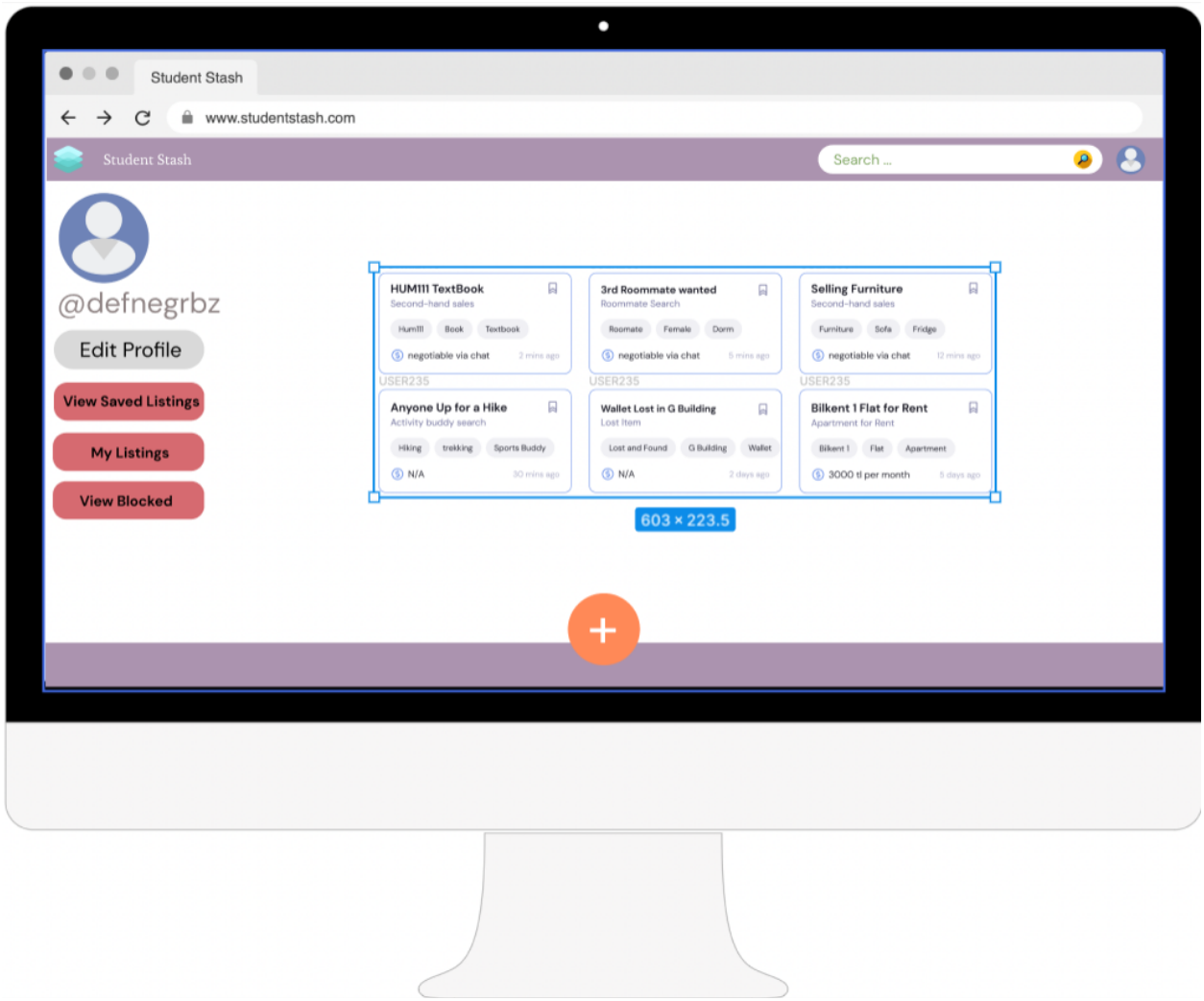
Found Listing



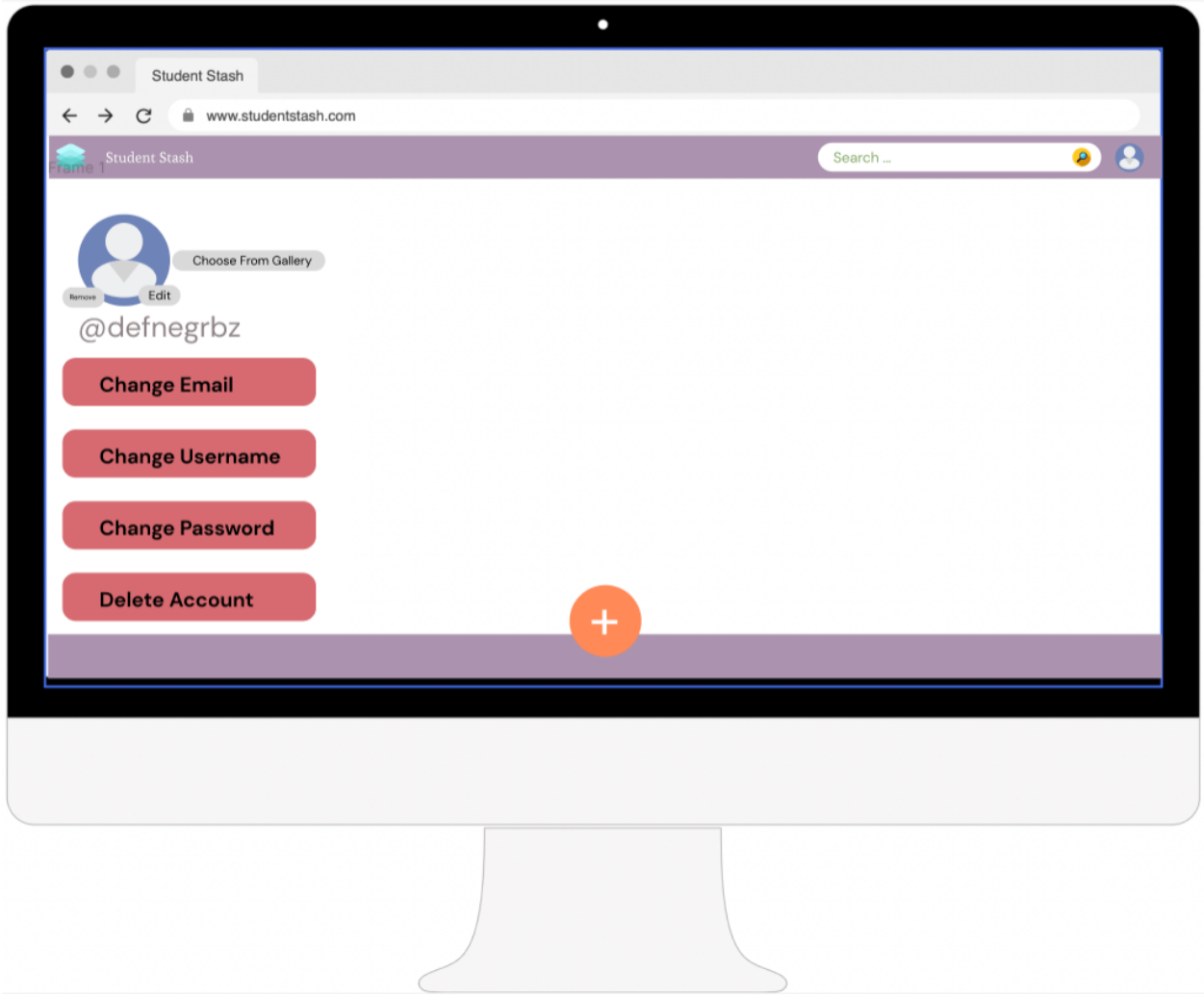
Activity Buddy Listing



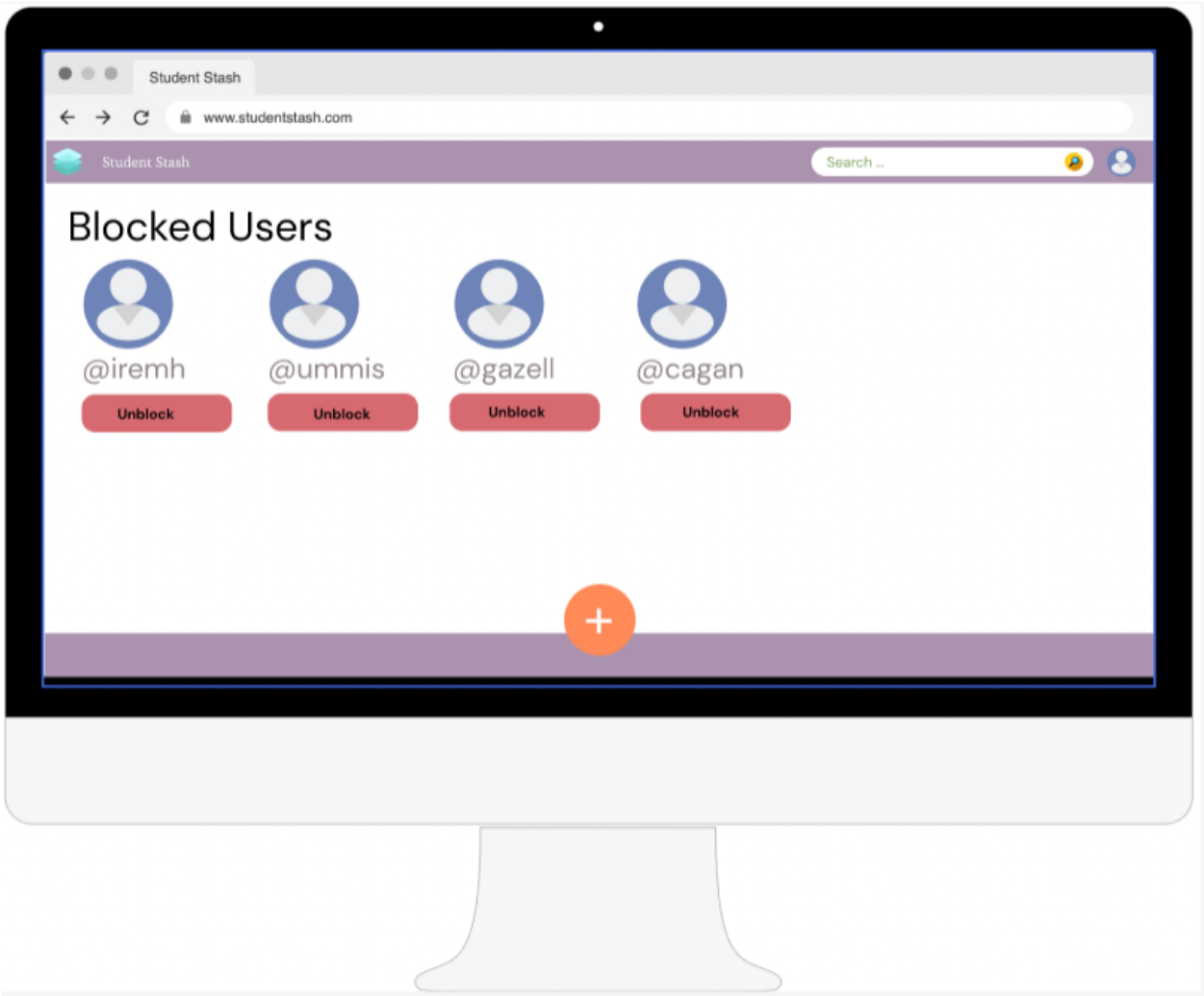
Your Profile Page



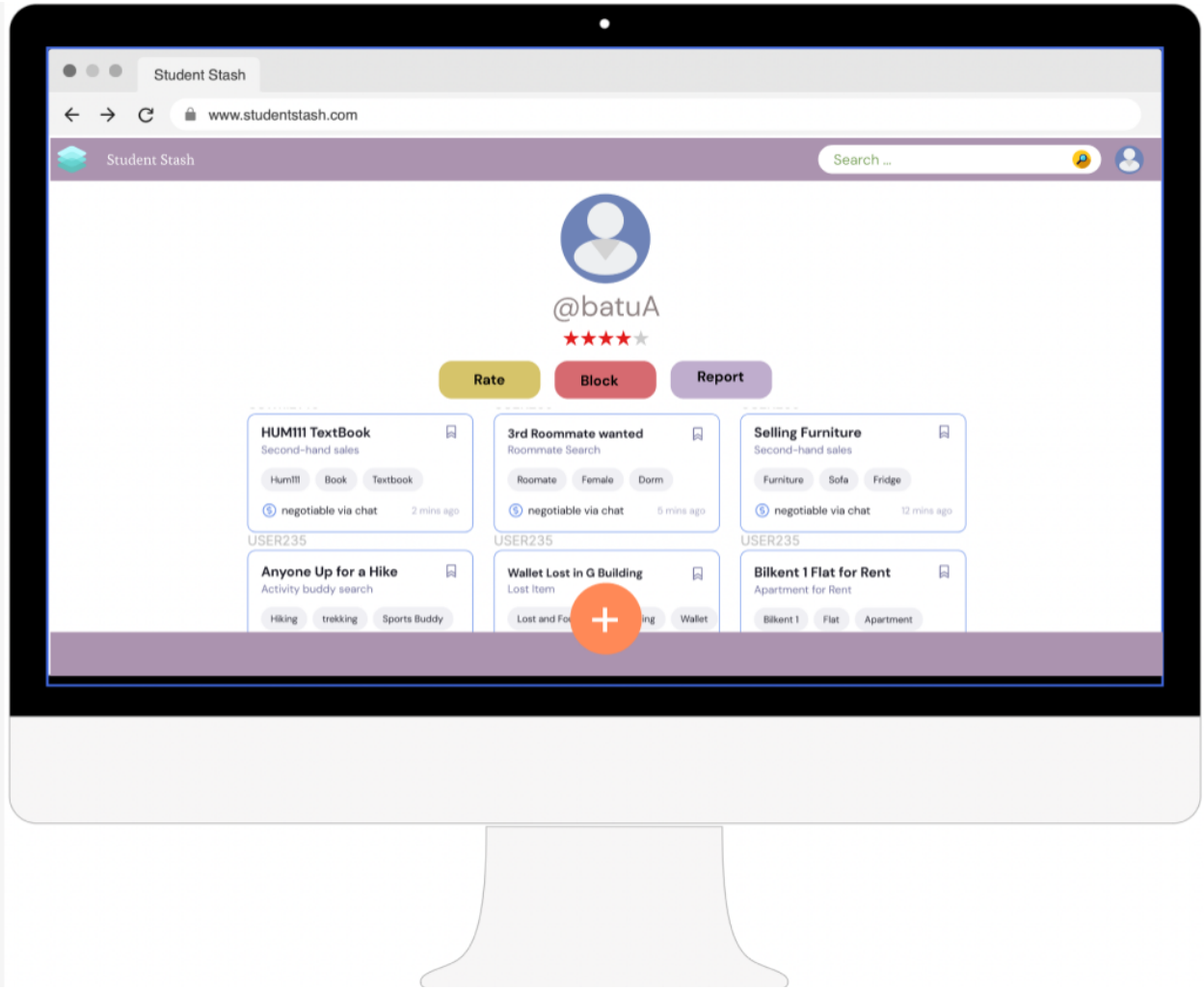
Edit Profile Page



Blocked Users Page



Other User’s Profile Page



View Chatbox