# Obtayn

Souvenir Exchange

A lot of times, there are region-specific authentic goods that we cannot always buy from e-commerce websites. The best way to procure these goods is to ask an acquaintance to bring them for you when they return. However, it is not necessary that every time we need something, we have someone to bring it for us.

To facilitate this exchange, we thought of a common platform where users post requests and the ones travelling can accept them. These requests can be viewed in the feed, along with photos of required items. Travellers can directly accept requests from here. Once a request gets accepted, the requestor & traveller can chat with each other.

## Salient Features

* **Post requests** - Users can request for a specific item by posting it on the ‘Feed’ and tag the location of procurement. (S)he can also include a picture of the item, if required.
* **Appreciation posts -** Users can post pictures of received items as a token of appreciation.
* **Feed** - Scrollable list containing pending requests. A traveller can accept requests directly from here. Also can be filtered by categories.
* **Requests Tab -** Users can view 2 categories of requests (incoming, outgoing).
* **Connecting People**- Users who accept requests from the feed can chat with the requestor for further communication.

**Assumption :** Once users establish direct contact, delivery & payment can be mutually organised by them.

## User Story + Description

Following User stories define Users’ interactions with the system, when s(he) opens our web application. They are written from the point of view of a person using our application, and written in a simple language.

1. [I, as a User, want to see the Sign Up form on the screen.](#_4zd4arxt1g60)
2. [I, as a User, want to see the Login form on the screen.](#_ezzn0ncncjmj)
3. [I, as a User, want to see the Homepage after login.](#_slt7ia6cdf46)
4. [I, as a Traveller, want to search for a Request on the Feed.](#_e4ziwohpo5qm)
5. [I, as a User, want to click on my username to view & edit my Profile details.](#_hbfi0m3vf28x)
6. [I, as a User, want to logout.](#_ys58os83ocwe)
7. [I, as a User, want to view all posts on the Feed.](#_h04855rmmqb1)
8. [I, as a User, want to filter the Feed to view only Requests or Appreciation Posts.](#_wgbubl50ckph)
9. [I, as a Traveller, want to accept a Request from the Feed.](#_npc6boli0jra)
10. [I, as a User, want to write a new Post.](#_v2epfyokz86h)
11. [I, as a Requestor, want to request a new item.](#_f2a307nxzhrn)
12. [I, as a Requestor, want to delete my request.](#_ubydxdwu0whx)
13. [I, as a User, want to delete my post.](#_jfuxeg1kv909)
14. [I, as a User, want to view Recent Activity (Notifications).](#_yrtsismnwayw)
15. [I, as a User, want to view all Requests.](#_69jcinl98e5x)
16. [I, as a User, want to view Chats.](#_1esqmi5l0iaa)
17. [I, as a User, want to send a message.](#_p3hfmtkt0bze)

### 

Following are User Stories mentioned above, with detailed description, task list, time estimate & test cases for each story.

For task estimations, we have considered **1 unit = 30min.**

### 

### Sign Up

|  |  |
| --- | --- |
| **1** |  |
| **User Story** | I, as a User, want to see the Sign Up form on the screen. |
| **Description** | Sign Up form contains Full Name, Email & Password fields (which are required), and Contact Number field as an optional field.  I can also choose to “Continue with Google”, where I can use my Google account to sign up.  After registration, the system shows a “Successful Registration” message, and I am redirected to the Login page. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a Signup component. | 4 |  |
| Create Sign Up form containing Name, email, password, confirm password & contact number fields. | 1 |  |
| Create “Continue with Google” button | 4 |  |
| Validate user inputs ,show error if invalid entries | 1 |  |
| Write a function to add user and validate email | 4 |  |
| Write a function to redirect to Login if Sign Up is successful. | 2 |  |
| **Total** | **16** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| To verify that Users do not mistype their email address while signing up. | System sends a confirmation link to the entered email address , which the user cannot confirm hence cannot login with this id. |  |
| To verify that Users do not Sign Up with pre-existing email id | System alerts Users that id is already in use. |  |
| To verify that Users do not sign up without entering all required fields | System notifies Users to fill the required fields . |  |

### Login

|  |  |
| --- | --- |
| **2** |  |
| **User Story** | I, as a User, want to see the Login form on the screen. |
| **Description** | If I am an existing User, I can log in using valid credentials (email & password), or I can choose “Continue with Google”.  After logging in I’m redirected to the Homepage |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a Login component. | 1 |  |
| Create Login form containing email & password fields, and a Login button. | 1 |  |
| Create “Continue with Google” button | 2 |  |
| Validate user inputs | 1 |  |
| Write a function to authenticate user | 4 |  |
| Write a function to authenticate Google user | 4 |  |
| Write a function to alert unregistered Users and block them from accessing the application. | 3 |  |
| Write a function to redirect to Homepage if Login is successful. | 2 |  |
| **Total** | **18** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| To verify that Users log in with valid credentials | System redirects users to the homepage. |  |
| To verify that Users do not log in with wrong password | System alerts the user that the password is incorrect |  |
| To verify that Users do not log in with unregistered email | System informs the user that the email is not registered. |  |
| To verify that Users log in with their Google account. | System redirects Users to Homepage if Google authenticates them. |  |

### Homepage

|  |  |
| --- | --- |
| **3** |  |
| **User Story** | I, as a User, want to see the Homepage after login. |
| **Description** | Given that I have logged in successfully, the system redirects me to Homepage containing the Feed(contains posts), header with options like Search Bar, Create Post, Open Chats, View Profile and link to navigate to Feed, Requests tab. This page also contains Recent Activity, which shows notifications. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create header component with   * Link to Feed page * Link to Requests page * Search bar * Create post button * Open chats button * Users’ name as a button | 4 |  |
| Create a “Create a Post” component with   * Pop up modal to add a post | 4 |  |
| Create a Feed component with   * Scrollable list of posts * “Back to Top” button | 1 |  |
| Create a Recent Activity component with   * Notification components | 1 |  |
| Write a function to fetch all required data into components. | 7 |  |
| Write a function to redirect to Requests from header. | 2 |  |
| Write a function to redirect to Homepage when User clicks on the system logo in the header. | 1 |  |
| **Total** | **20** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| To verify that Users do not access homepage without logging in | The system blocks them from going forward, and redirects Users to the login page. |  |
| User is on Homepage | The system shows the user's name and loads all the components. |  |
| User is on the Homepage, and clicks on “Feed” in the header. | System refreshes Homepage. |  |
| User clicks on the system logo in the header. | System redirects User to Homepage. |  |

## 

## 

### Search

|  |  |
| --- | --- |
| **4** |  |
| **User Story** | I, as a Traveller, want to search for a Request on the Feed. |
| **Description** | On the Homepage, the system shows a “Search” section in the Header.  This section contains a drop-down list of Categories (Clothing, Home & Living, Hobbies, Electronics, Food), a search box for keywords, and a location-picker which autocompletes. After entering the required inputs and clicking on the Search button, the system displays posts corresponding to the search, which helps to filter the posts. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a search component, containing a dropdown list of categories, textbox for keyword, and a location picker. | 6 |  |
| Load all the categories to show in dropdown. | 1 |  |
| Write a function to show relevant requests as search results. | 4 |  |
| Write a function to handle “Network Error” exception. | 2 |  |
| Write a function to handle “No match found” exception. | 2 |  |
| **Total** | **15** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| To verify that Users do not click the Search button without any data input i.e. no Category selected, no Keyword, no Location picked. | System should alert Users to enter data into at least one field. |  |
| System does not find any posts that match Users’ search. | System should display a message - “Sorry, we could not find any matches for your search”. |  |
| Network error occurs while Users are searching for a Request. | System alerts Users of the issue, and display results when connection is established. |  |
| A particular Request appears in Users’ search result and the User tries to accept it, while it is simultaneously getting accepted by someone else. | System alerts the User - “Thank you for your interest, but this request is already accepted by someone.” |  |
| To verify that Users do not enter invalid location | System prompts User to enter a valid location. |  |

### Profile

|  |  |
| --- | --- |
| **5** |  |
| **User Story** | I, as a User, want to click on my username to see my Profile details. |
| **Description** | When I click on my Username on the top-right corner of any page, the system shows a small window with my name, email and profile picture. Profile picture has an “Edit” button beside it. On clicking this button, the system shows options to choose a new photo to set as my profile picture. I can only upload photos up to 1Mb of size.  At the bottom of this window, I have the option to Log out. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a Profile component containing -   * profile picture * Edit profile picture option * User name * Email * Logout button | 4 |  |
| Write a function to display the Profile component when the User’s name is clicked. | 2 |  |
| Write a function to load Profile details into the pop-up window. | 3 |  |
| Write a function to check if a newly selected image for profile fits within constraints (extension, size). | 2 |  |
| Write a function to update profile picture | 4 |  |
| **Total** | **15** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| To verify that Users do not select an invalid file type for profile photo. | System should alert Users of valid file formats and prompt Users to choose another image. |  |
| To verify that Users do not select an image that is too big in size, as a profile photo. | System should inform Users of the maximum image size that is accepted, and prompt Users to choose another image. |  |
| Users click on their name in the top left corner of the header. | System shows a pop-up window with profile details. |  |
| Users click outside the Profile pop-up window. | System closes the pop-up. |  |
| Users update their profile photo. | System refreshes the pop-up window, and displays the updated profile photo. |  |

## 

### 

### Logout

|  |  |
| --- | --- |
| **6** |  |
| **User Story** | I, as a User, want to log out of the application. |
| **Description** | When I click on my Username on the top-right corner of any page, the system shows a small window with my name, email and profile picture.  At the bottom of this window, I have the option to Log out. |

## 

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Write a function to redirect User to the Login page when Logout is clicked. | 2 |  |
| **Total** | **2** |  |

## 

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Users click on Logout. | System redirects Users to Login page. |  |
| Network error occurs while Users are logging out. | System alerts Users of the issue, and logs Users out when connection is established. |  |

## 

### Feed

|  |  |
| --- | --- |
| **7** |  |
| **User Story** | I, as a User, want to view all posts on the Feed. |
| **Description** | On the Homepage of the system, I can view all posts in the Feed.  The Feed contains 2 types of posts - Requests and Appreciation Posts. I can scroll through these posts. To go back to the top of the Homepage, I can click the “Back to Top” button which appears on the bottom-right corner of the screen. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a Post component which contains   * User name * User profile pic * Title * Timestamp * Message * Images(if any) 5 max | 4 |  |
| Create a Request component which contains   * User name * User profile pic * Title * Message * Timestamp * Location * Category * Accept button | 4 |  |
| Create a Filter component which will have a dropdown with 2 options - Requests and appreciation posts | 1 |  |
| Create a Feed component which contains Filter & multiple Post and Request components | 5 |  |
| Create a “Back to Top” button. | 1 |  |
| Write a function to make the “Back to Top” button visible after User scrolls a particular number of posts. | 1 |  |
| Write a function to go back to top of the page when “Back to Top” button is clicked | 1 |  |
| Write a function to fetch all Posts and use the data for the Post component. | 2 |  |
| Write a function to fetch all pending requests and use it for Request component | 2 |  |
| Write a function to store all posts and requests in array of objects, sorted by timestamp | 4 |  |
| Write a function to load all the Feed components. | 2 |  |
| **Total** | **27** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Users open Homepage and Feed is empty | System displays a message - “We’re all caught up! No recent posts.” |  |
| Users click on an image in the Feed. | System maximizes the image in a pop-up modal. |  |
| A User posts on the Feed. | System refreshes Feed for all Users. |  |

## 

### Filter

|  |  |
| --- | --- |
| **8** |  |
| **User Story** | I, as a User, want to filter the Feed to view only Requests or Appreciation Posts. |
| **Description** | At the top of the Feed, I can select to view only Requests or only Appreciation Posts. On selecting any one filter, the Feed shows only that type of posts.  When viewing only requests the system asks for my location by asking for location permission (or I enter it manually) and sorts Request posts according to the ones nearest to me. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create filter options - #request & #appreciation. | 1 |  |
| Write a function to fetch the selected option. | 2 |  |
| Write a function to filter the feed by only showing selected option | 4 |  |
| Write a function to be called when #request is selected   * Asks user for location permission * Asks user to manually enter location if permission is denied * Store user location | 6 |  |
| Write a function that takes requests location and user location and sorts the array by nearest to farthest | 5 |  |
| **Total** | **18** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| User clicks on the filter dropdown menu. | The dropdown menu shows the filters available |  |
| User selects a filter | The Feed is refreshed with only the filtered posts |  |
| User selects #request filter | The system asks the User for location access or the User enters the location manually |  |
| The Feed is sorted by location nearest to farthest from the User’s location. |  |

### 

### Accept Req from Feed

|  |  |
| --- | --- |
| **9** |  |
| **User Story** | I, as a Traveller, want to accept a Request from the Feed. |
| **Description** | When I scroll through Request posts, I see an option to accept the Request. On clicking this option, System prompts me to confirm my action. When I click “Continue”, System shows “Request Accepted” message and redirects me to the Requests page. Under “Incoming Requests” the current request is highlighted. The User is added to my Chat list, and we can start a new conversation. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Make a function to change the status of the Request on click of accept button. | 2 |  |
| Make a function to handle cases of already accepted requests. | 1 |  |
| Make a function which adds a notification. | 2 |  |
| Make a function to add Users to the Chat list. | 3 |  |
| Make a function to redirect to the Requests page. | 2 |  |
| Write a function to highlight current Request. | 2 |  |
| **Total** | **12** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| User accepts a Request from the feed. | The request is removed from the Feed, and added to his/her incoming requests. |  |
| The requestor is added to the User's chat list. |  |
| The system sends a notification to the requestor. |  |
| Network error occurs while the User is accepting a Request from the Feed. | System notifies the User about the issue, and changes Request status as soon as connection is established. |  |
| To verify that two Users do not accept a Request simultaneously. | System will compare timestamps and allocate the request to the User who accepted it first. |  |
| System refreshes Feed for all other Users, and the corresponding Request is removed from the Feed. |  |
| To verify that Users do not accept their own Requests. | System blocks the User from accepting their own Request. |  |

### Create Post

|  |  |
| --- | --- |
| **10** |  |
| **User Story** | I, as a User, want to create a new Post. |
| **Description** | When I click on the “Create Post” button in the header, or the one above the Feed, I see a pop-up window. When I click on “Tag”, I see a drop-down list containing Request, Post and I select “Post”.  This window contains a form with various fields like Title, Message and Photo.  On clicking the “Add Post” button, the pop-up window closes and my Post appears at the top of the Feed. System shows my Username and Profile picture on the left of the post title. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Write a function to trigger modal on click for “Create a Post” component or create post button in header | 2 |  |
| Make a dropdown component to select Tag (request post or appreciation post) | 2 |  |
| Make a component for appreciation post form containing :   * Title * Message * images(if any, max 3) * Add Post button | 3 |  |
| Make a function to validate image size and extension | 2 |  |
| Write a function to show request form or appreciation form depending on Tag | 1 |  |
| Write a function to add the new appreciation post on click of Post button | 2 |  |
| Write a function to close the Create Post modal when the “Add Post” button is clicked. | 1 |  |
| Write a function to show the new Post on top of the Feed. | 4 |  |
| Write a function to handle “Network Error” exception. | 2 |  |
| **Total** | **19** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| User clicks “Create Post” button | System shows the Create Post modal. |  |
| User clicks the “Add Post” button. | The post is added to the database and to the Feed. |  |
| To verify that Users do not keep required fields are empty.  (title, message) | System prompts Users to fill fields that are marked required. |  |
| User clicks outside the modal or clicks close button | The modal is closed |  |
| Network error occurs while User is uploading the new Post | System uploads the Post as soon as connection is established. |  |
| User enters data in Create Post form, and closes the pop-up before clicking on Add Post. | System resets form fields to blank. |  |
| User enters data in Create Post form, and toggles Tag before clicking on “Add Post”. | System resets form fields to blank. |  |

### Create Request

|  |  |
| --- | --- |
| **11** |  |
| **User Story** | I, as a Requestor, want to request a new item. |
| **Description** | When I click on the “Create Post” button in the header, or the one above the Feed, I see a pop-up window. When I click on “Tag”, I see a drop-down list containing Request, Post and I select “Request”.  This window contains a form with various fields like Title, Message, Photo, Location and Category. Photo is an optional field. I cannot upload more than 3 photos.  On clicking the “Add Request” button, the pop-up window closes and my Post appears at the top of the Feed. System shows my Username and Profile picture on the left of the post title. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Use “Create a Post” component from story #7. | 0 |  |
| Make a component for request form containing :   * Title * Message * Location * Category * images(if any, max 3) * Add Request button | 5 |  |
| Write a function to trigger the modal on click for Create Post component. | 1 |  |
| Write a function to validate whether form fields are filled. | 1 |  |
| Write a function to show request form or appreciation form depending on Tag | 0 |  |
| Write a function to validate image constraints. | 2 |  |
| Write a function to add the new Request when the Post button is clicked. | 2 |  |
| Write a function to show the new Request on top of the Feed. | 2 |  |
| Write a function to handle “Network Error” exception. | 2 |  |
| Write a function to close the Create Post modal when the “Add Request” button is clicked. | 0 |  |
| **Total** | **15** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| To verify that Users do not keep required fields are empty.  (title, message, tag, location, category) | System prompts Users to fill fields that are marked required. |  |
| Network error occurs while User is uploading his Request. | System uploads the Request as soon as connection is established. |  |
| User clicks on Create Post button | System opens a pop-up modal, containing a form to add a new post. |  |
| User clicks anywhere outside the pop-up modal. | System closes the modal. |  |
| User enters data in Create Post form, and closes the pop-up before clicking on Post. | System resets form fields to blank. |  |
| User clicks the “Add Request” button. | The post is added to the database and to the Feed. |  |
| User enters data in Create Post form, and toggles Tag before clicking on “Add Request”. | System resets form fields to blank. |  |

### Delete a request

|  |  |
| --- | --- |
| **12** |  |
| **User Story** | I, as a Requestor, want to delete my Request. |
| **Description** | When I scroll through Request posts, I see an option to delete my Request. On clicking this option, System prompts me to confirm my action (“Are you sure?”). Once I click continue , the system deletes the request and shows a message saying “Request Deleted” and refreshes the Feed. In the Requests page, under “Outgoing Requests” I can see a “Cancel” button. Once I click the button, the system asks me to confirm my action. When I click confirm, the system deletes the request and sends a notification to the traveller, and redirects me to the requests page. |

|  |  |
| --- | --- |
| Task | Estimated Units |
| Write a function to show a confirm box when the delete icon is pressed. | 2 |
| Write a function to delete the request from the database. | 2 |
| Write a function to add a “Request Deleted” notification. | 2 |
| Write a function to refresh the Feed page. | 1 |
| Write a function to redirect User to the Request page | 1 |
| Write a function to show a confirm box when the “Cancel” button is pressed. | 2 |
| Write a function to remove the request from the requestor's “Outgoing” and travellers “Incoming” request. | 2 |
| **Total** | **12** |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| User deletes a Request from the feed. | The system shows a confirm window. |  |
| The request is removed from the Feed, and from his/her outgoing requests. |  |
| Network error occurs while User is deleting a Request from the Feed. | System notifies the User about the issue, and deletes Request as soon as connection is established. |  |
| User deletes a pending request from the Request page. | The system shows a confirm window. |  |
| The request is removed from the Outgoing Requests & Feed. |  |
| User deletes a request which was accepted. | The system shows a confirm window. |  |
| The request is removed from the outgoing requests. |  |
| The request is deleted from the traveller’s “incoming request”. |  |
| The system sends the Traveller a notification saying “ Request deleted”. |  |

## 

## 

### Delete a post

|  |  |
| --- | --- |
| **13** |  |
| **User Story** | I, as a User, want to delete my Post. |
| **Description** | When I scroll through posts on the Feed, I see an option to delete my Post. On clicking this option, System prompts me to confirm my action (“Are you sure?”).  When I click continue , the system deletes the post, shows a message saying “Post Deleted” and refreshes the Feed. |

|  |  |
| --- | --- |
| Task | Estimated Units |
| Write a function to show a confirm box when the delete icon is pressed. | 2 |
| Write a function to delete the post from the database. | 2 |
| Write a function to refresh the Feed page. | 1 |
| Write a function to show a confirm box when the “Cancel” button is pressed. | 1 |
| **Total** | **6** |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| User deletes a Post from the feed. | The system shows a confirm window. |  |
| The Post is removed from the Feed. |  |
| Network error occurs while User is deleting a Post from the Feed. | System notifies the User about the issue, and deletes Post as soon as connection is established. |  |

### 

### Recent Activity

|  |  |
| --- | --- |
| **14** |  |
| **User Story** | I, as a User, want to view Recent Activity (Notifications). |
| **Description** | When I open the Homepage section, I see the “Recent Activity” section on the right of the screen, which shows 5 most recent notifications sorted from most recent to least recent.  When I click on a **Chat** notification bubble (which shows new message alerts), the system redirects me to that particular chat screen. When I click on a **Request** notification (i.e. when someone has accepted my Request), the system redirects me to the Requests page. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Use the “Recent Activity” component form user story #3 and add style properties. | 2 |  |
| Write a function to fetch notifications for current User. | 4 |  |
| Write a function to display fetched Activity objects in “Recent Activity” sorted by timestamp. | 3 |  |
| Create chat notification component (clickable)   * Write a function to redirect to the chat when User clicks on it. | 4 |  |
| Create a request update notification component (clickable)   * Write a function to redirect to a Requests page. * Highlight the corresponding Request. | 6 |  |
| Write a function to refresh Recent Activity once new notifications arrive. | 2 |  |
| **Total** | **21** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Users click on a notification. | System redirects Users to the respective page/section, depending on type of notification clicked. |  |
| Users receive a notification while they are on the Recent activity section. | System refreshes Recent Activity to display the new notification at the top, and deletes the oldest notification. |  |

## 

### View all Requests

|  |  |
| --- | --- |
| **15** |  |
| **User Story** | I, as a User, want to view all Requests. |
| **Description** | Clicking in the ‘Requests’ tab on the header redirects me to the Requests page. Here requests are categorized into “Incoming” & “Outgoing”.  Requests that I have accepted are listed under Incoming, while Requests posted by me are listed under Outgoing. Under Outgoing requests, I can mark an accepted request as Fulfilled, once I receive the requested item. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a container with Incoming (Requests accepted by the User) & Outgoing (Requests posted by the User) requests. | 2 |  |
| Create Incoming requests component with details - requestor, title, location, message, images(if any). | 4 |  |
| Create Outgoing requests component with details - acceptor, title, location, message, images(if any), Fulfilled button(only for accepted requests),cancel button | 4 |  |
| Write a function to fetch all the Requests related to the user. | 3 |  |
| Write a function to display all the fetched Requests. | 5 |  |
| Write a function to differentiate Outgoing requests depending upon status(accepted/pending). | 2 |  |
| Write a function to refresh status of Outgoing requests (from pending to accepted). | 3 |  |
| Write a function to handle Fulfilled button click - show prompt to confirm whether User wants to continue, delete Request if yes. | 5 |  |
| **Total** | **28** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Status of Outgoing Request changes while User is on the Requests page. | System refreshes the page and updates Request status. |  |
| To verify that an Outgoing Request is accepted by someone. | System displays the Request in a different color, to differentiate it from other pending requests. |  |
| Users click on the “Fulfilled” button on an accepted request. | System prompts Users to confirm whether they want to continue. |  |
| Users click on “Continue” in the Fulfilled button prompt. | System deletes the corresponding Request permanently. |  |

### View Chat List

|  |  |
| --- | --- |
| **16** |  |
| **User Story** | I, as a User, want to view Chats. |
| **Description** | On clicking the “Chats” button in the header, the system redirects me to the Requests tab, which also shows recent chats. I can see a list of chats, with the sender’s profile picture, name and a preview of the most recent message in that thread. Unread messages are shown in bold text. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Create a Chats component to the right of Requests. | 2 |  |
| Write a function to redirect to the Requests page when the User clicks on the Chats button (available on the header) | 2 |  |
| Write a function to fetch all User Chats. | 4 |  |
| Write a function to group messages by different users | 3 |  |
| Write a function to fetch user profiles and names. | 3 |  |
| Write a function to take all the data and make a separate Chat component containing -   * Sender’s profile picture * Sender’s name * last message * timestamp of the last message. | 10 |  |
| Write a function to highlight unread chat in bold. | 1 |  |
| **Total** | **25** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Users have new & unread messages. | System displays the message preview in bold. |  |
| Users’ Chats exceed the maximum number of Chats that can be displayed on one screen. | System allows Users to scroll through all Chats. |  |

### View Chat thread

|  |  |
| --- | --- |
| **17** |  |
| **User Story** | I, as a User, want to send a message. |
| **Description** | When I click on any thread in Chats, the system shows me a chat screen.  On this screen, I can read previous chats, send a new message and add photos using the ‘Add’ button. On clicking the Send button, my message appears in the thread. I can select only 3 images at a time.  When I click on the Back button, the system redirects me back to Chats. |

|  |  |  |
| --- | --- | --- |
| Task | Estimated Units | Actual Units |
| Write a function to switch from the chat window to the chat thread with a particular User. | 2 |  |
| Write a function which uses data fetched in Story #13, and display the corresponding message thread. | 3 |  |
| Make a function to display the message thread, such that one can differentiate sender’s and receiver’s messages, with timestamps. | 5 |  |
| Write a function to scroll down on a new message. | 2 |  |
| Create a component which contains a text field for text message, file selector for attaching images and a button to send the message. | 6 |  |
| Create a back button which redirects Users to a list of chats. | 2 |  |
| Write function to handle “Network Error” exception. | 2 |  |
| Write function to check whether selected image fits within constraints (size, file type, number of images selected) | 2 |  |
| **Total** | **26** |  |

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Users click on a Chat notification. | System redirects User to chats section, and opens the corresponding thread with the sender. |  |
| Network error occurs before Users’ message is sent. | System should alert Users of the issue, and send messages as soon as connection is established. |  |
| To verify that Users do not send more than 3 images at a time. | System should alert User about the image limit. |  |
| Users want to view the entire Chat. | System allows Users to scroll through the entire message thread. |  |

## 

## Other Tasks

**1 unit = 30min**

### Estimation & Planning

|  |  |  |
| --- | --- | --- |
| Task | Estimated Time  Must have/Good to have | Actual Time |
| Identify User Stories | 2/2 | 4/2 |
| Write one-liners for User Stories | 6/3 | 7/3 |
| Write descriptions for each User Story | 8/2 | 8/2 |
| Write detailed task list for each User Story | 16/6 | 14/4 |
| Write test cases for each User Story | 8/3 | 11/3 |
| Peer review | 8/3 | 10/4 |
| **Total** | **48 /19** | **54/18** |

### Requirement Analysis

|  |  |
| --- | --- |
| Task | Actual Time |
| Gathering Requirements | 20 |
| Categorize requirements according to MOSCOW | 5 |
| Validation of requirements | 12 |
| Refinement of requirements | 32 |
| **Total** | **69** |

### Design

|  |  |
| --- | --- |
| Task | Actual Time  Must have/Good to have |
| Database Design   * Identify entities * Identify attributes for each entity * Identify relationships between entities | 16/3 |
| ERD | 4/2 |
| Components   * Identify Frontend components * Identify APIs * Identify Backend components | 2/0  2/1  1/0 |
| Draw Component diagram | 2/1 |
| Draw rough screen designs with pen & paper | 4/1 |
| Recreate screens in draw.io | 5/0 |
| Learn basics of Adobe XD | 3/0 |
| Create wireframes in Adobe XD | 10/5 |
| Prototyping | 4/0 |
| Use Cases | 3/0 |
| **Total** | **56/13** |

### Documentation

|  |  |  |
| --- | --- | --- |
| Task | Estimated Time | Actual Time |
| Create Requirements document containing   * Introduction * Salient features * MOSCOW requirements | 2 | 2 |
| Write Use cases | 2 | 3 |
| Create Design document containing   * List of entities & their descriptions * ERD * Component diagram * Wireframes | 5 | 6 |
| Add snapshots of the system & add their description | 4 | 4 |
| Add test cases | 5 |  |
| Execute test cases | 20 |  |
| Write Limitations & Future Enhancements | 2 |  |
| Write a Conclusion | 2 |  |
| **Total** | **42** |  |

### 

### Setup/Learning

|  |  |  |
| --- | --- | --- |
| Task | Estimated Time | Actual Time |
| Get familiar with Firebase & write POCs   * Authentication * Storage * Hosting | 32 |  |
| Get familiar with Firestore & write POCs | 16 |  |
| Get familiar with Redux | 16 |  |
| React external libraries | 16 |  |
| **Total** | **80** |  |

**Total ->** 327 units ≅ 164 hrs ≅ 21 days

## Total time estimated

**Development** -> 292 units = 146 hours ≅ 19 days (considering 8 hour work days)

* Must Haves

**Development** -> 78 units = 34 hours ≅ 5 days (considering 8 hour work days)

* Good to Haves

**Other Tasks (Requirement Analysis, Design, Documentation, Setup/Learning)** ->

327 units ≅ 164 hours ≅ 21 days

|  |  |  |
| --- | --- | --- |
|  | Total Estimate (hrs) | Total Estimate (days approx) |
| Must Have Only | 146+164 = 310 | 19+21 = 40 |
| Must Have + Good to Have | 180+164 = 344 | 24+21 = 45 |

## 

## 

## Requirement Analysis

**Must have requirements**

1. Users Profile (name, email, phone no, profile pic)
2. Feed containing requests, appreciation posts
3. Filters to categorize posts on the feed - Apparel, Lifestyle, Electronics etc
4. Section to view requests (Incoming, Outgoing - both containing Pending & Accepted requests)
5. Functionality to accept requests
6. Notification when someone accepts a User’s request
7. Functionality to chat with users on request acceptance
8. Feed can be sorted by current location
9. Users can allow location access or manually set current location
10. Users can mark an accepted request as “Fulfilled”, which suggests that the User has received their item. This will permanently delete that request from the system.

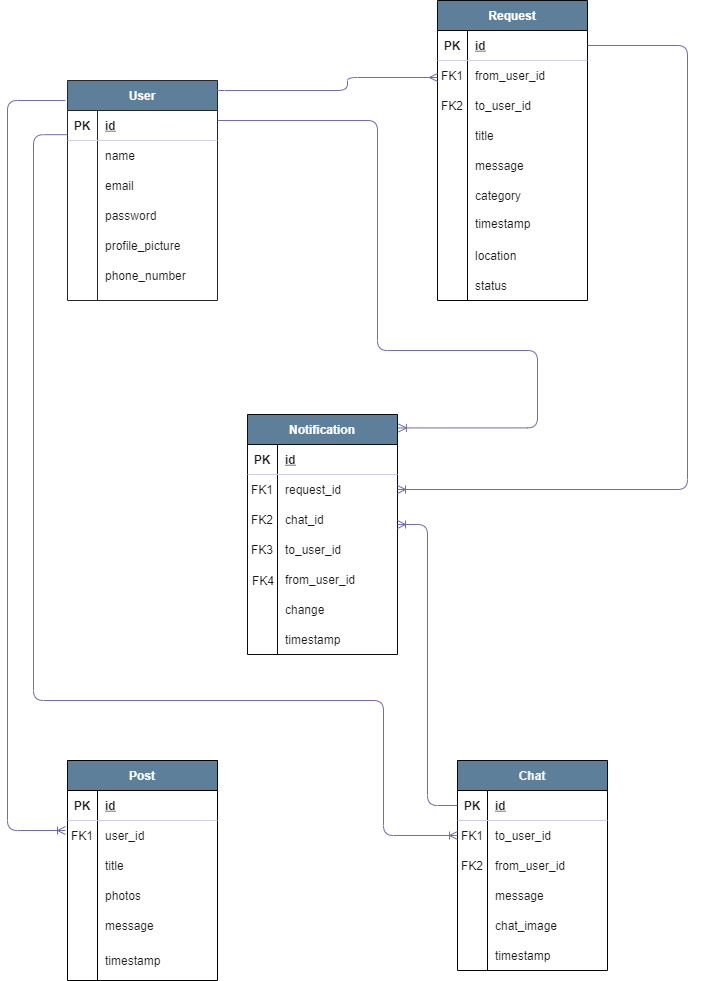
[**Nice to have requirements**](https://docs.google.com/document/d/1266fi-ZnlW3J73LJMnkBEcxAYfohuAmND62SO-qy9gg/edit?usp=sharing)

1. Traveller posting available items
2. Requestor can send targeted request

## Database Design

### Entity Relationship Diagram

* This diagram shows relationships between the entities in our system.
* One User can post many Requests, create many Posts, have many Chats and view many Notifications. Hence, all these are One-Many relationships.
* Notification is a many-many relationship which maps Users, their Requests (or Chats) to each other. Hence, it contains user\_id, request\_id & chat\_id as foreign keys.



### Entity Description

**USER**

This database table stores User details. Each row is identifiable via the id, which is used as primary key and is unique for each User. Profile\_picture & phone\_number are optional fields for Users.

|  |  |
| --- | --- |
| Attribute | Constraints |
| id | PRIMARY KEY |
| name | NOT NULL |
| password | NOT NULL |
| profile\_picture |  |
| phone\_number |  |

## 

**REQUEST**

This table represents a Request posted by User(s). “to\_user\_id” is NULL by default, and it changes only when that Request is accepted by another User.

Each Request has a title (identifies a Request), message (requirement description written by the User), category (Clothing, Home & Living, Hobbies, Electronics, Food), timestamp (date & time when the Request is posted), location (the place where User)

|  |  |
| --- | --- |
| Attribute | Constraints |
| id | PRIMARY KEY |
| to\_user\_id | NOT NULL |
| from\_user\_id | NOT NULL |
| title |  |
| message |  |
| category |  |
| timestamp |  |
| location |  |
| photos |  |
| status |  |

**POST**

Represents who(user\_id) has posted what(title, photos, message) on the Feed. Also stores the time when it was posted (timestamp). Users may or may not add photos to their posts, so it is an optional field.

|  |  |
| --- | --- |
| Attribute | Constraints |
| id | PRIMARY KEY |
| user\_id | NOT NULL |
| title |  |
| message |  |
| timestamp |  |
| photos |  |

**CHAT**

Represents each message that Users send to each other. It contains 2 foreign keys - sender (from\_user\_id) & receiver (to\_user\_id). Users can send just a message (chat\_image is NULL) or Users can directly send an image without any message (message), which makes them optional fields.

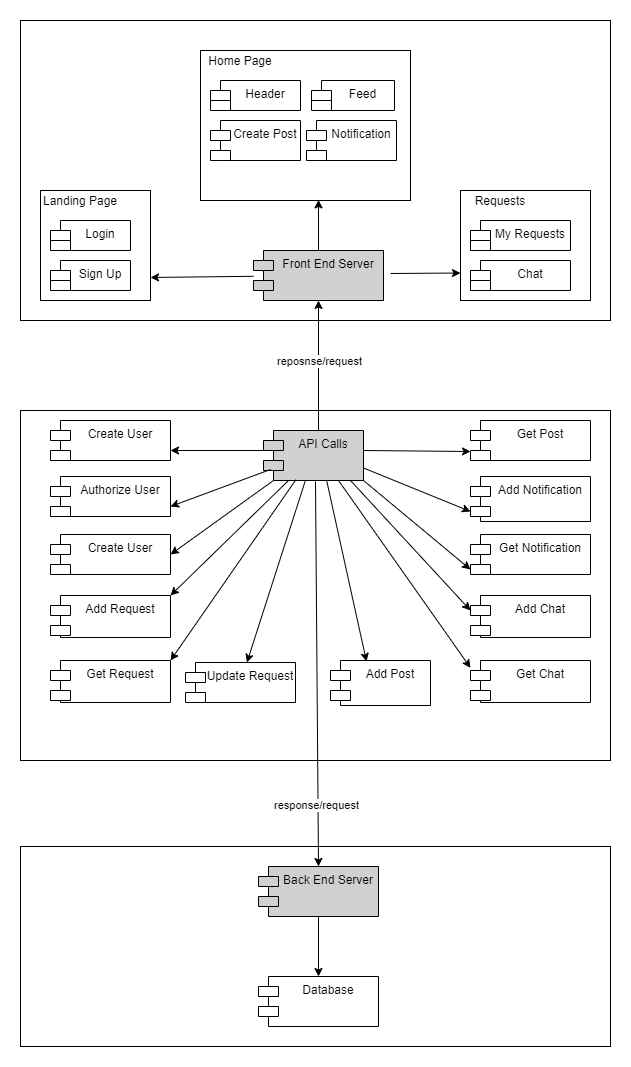
|  |  |
| --- | --- |
| Attribute | Constraints |
| id | PRIMARY KEY |
| from\_user\_id | NOT NULL |
| to\_user \_id |  |
| message |  |
| timestamp |  |

**NOTIFICATION**

A Notification can be triggered by 2 events - User receives a message from another User (to\_user\_id, message) or User’s request is accepted by another User (request\_id, change). Since notifications are temporary, it also stores the time when notification appears (timestamp).

|  |  |
| --- | --- |
| Attribute | Constraints |
| id | PRIMARY KEY |
| trigger\_event | NOT NULL |
| trigger\_event\_id | NOT NULL |
| from\_user\_id | NOT NULL |
| to\_user \_id | NOT NULL |
| message |  |
| timestamp |  |

## Component Diagram



## Technology Stack

## 

|  |  |
| --- | --- |
| Technology | Reasons |
| React (v17.0.2) | * Already familiar with the library. * Vast resources available online. * Helps to build interactive & rich User Interfaces. |
| Firebase - Hosting Services | * Provides ready-to use environment that allows to deploy code fast |
| Firebase - Backend Services | * Free authentication, database, storage & hosting. |
| React Redux (v7.2.3) -Application State Management | * Supports React v17 |
| Firestore - Online Databases (NoSQL) | * NoSQL cloud database * Easy integration with firebase app |

## 

## 

## Coding Guidelines

* Use prettier.io
* Use functional components ONLY
* PascalCase for function names, file names
* camelCase for variables, try to avoid vague names
* For constants : MY\_CONSTANT
* Tab length : 4 spaces
* Add function description before each component - purpose, arguments, return value
* Convention for hooks : useHooks
* Functions return 0 for error

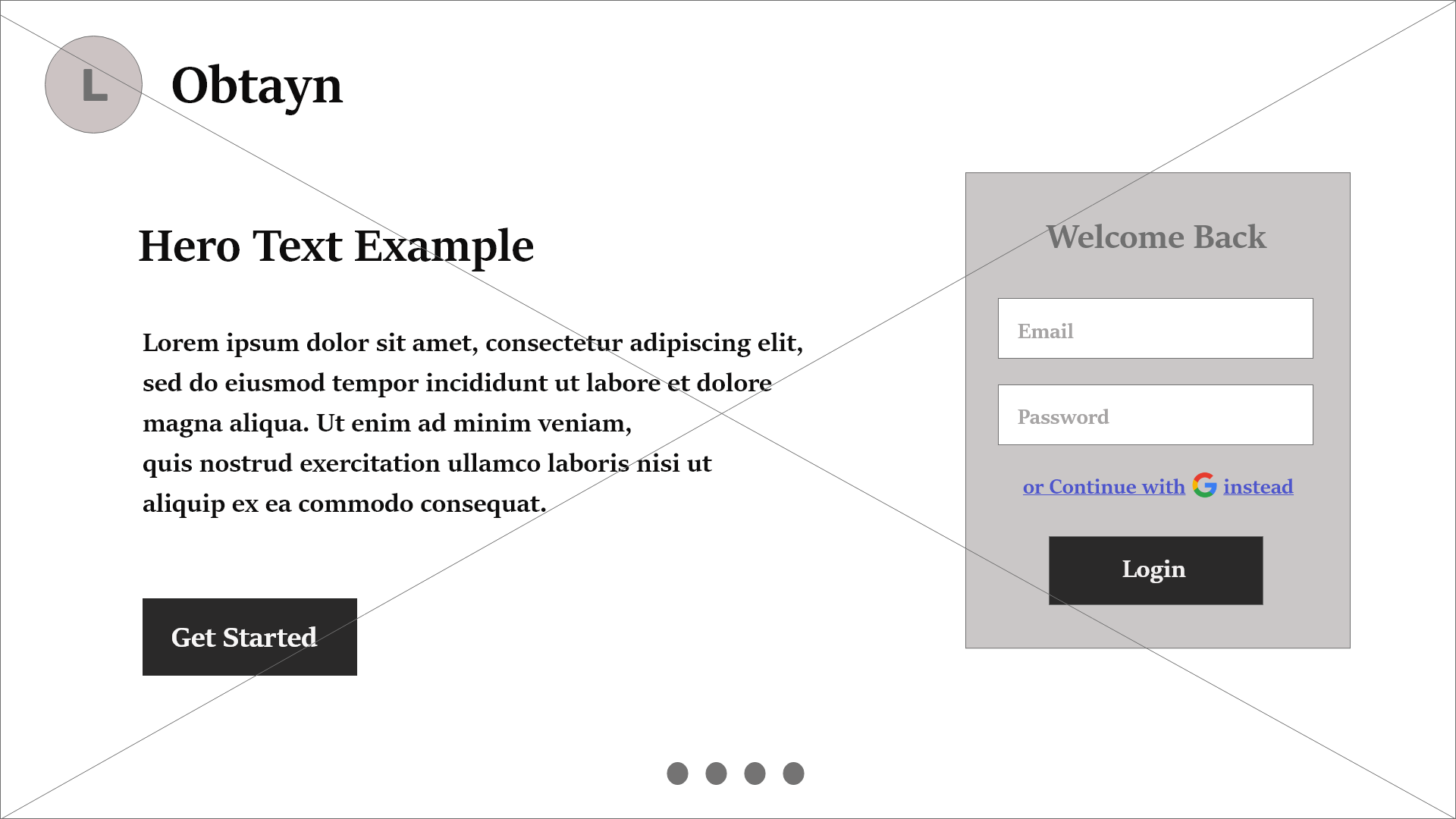
## 

## Screen Mockups

Protoype link - <https://xd.adobe.com/view/1fe60dd3-b151-482e-bbdf-b6f93b6e30eb-4ab5/>

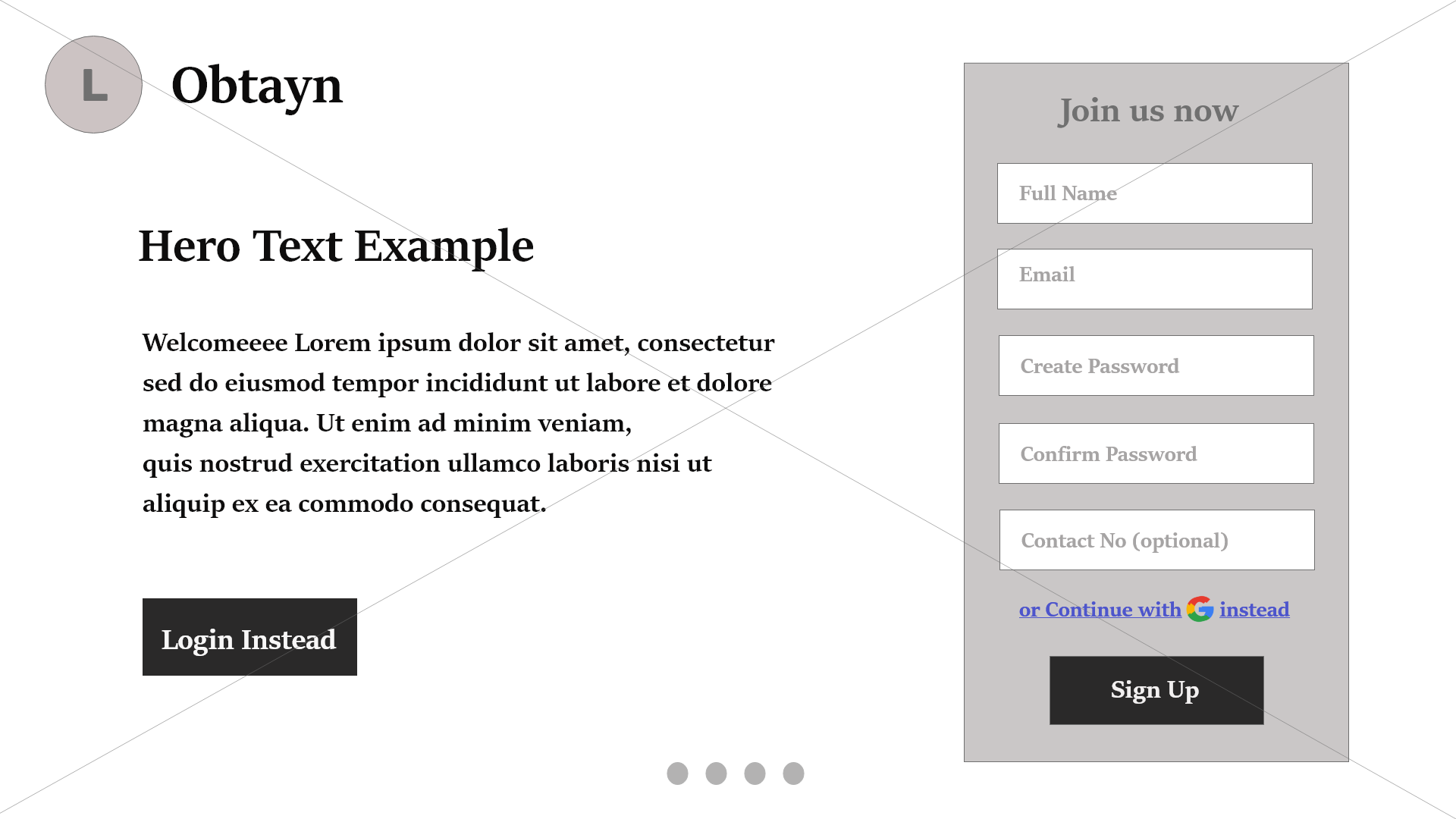
### Landing Page - Login

This is the first page that Users are directed to. It contains a short description of the application. Existing users can Login directly or click on ‘Get Started’ to sign up.



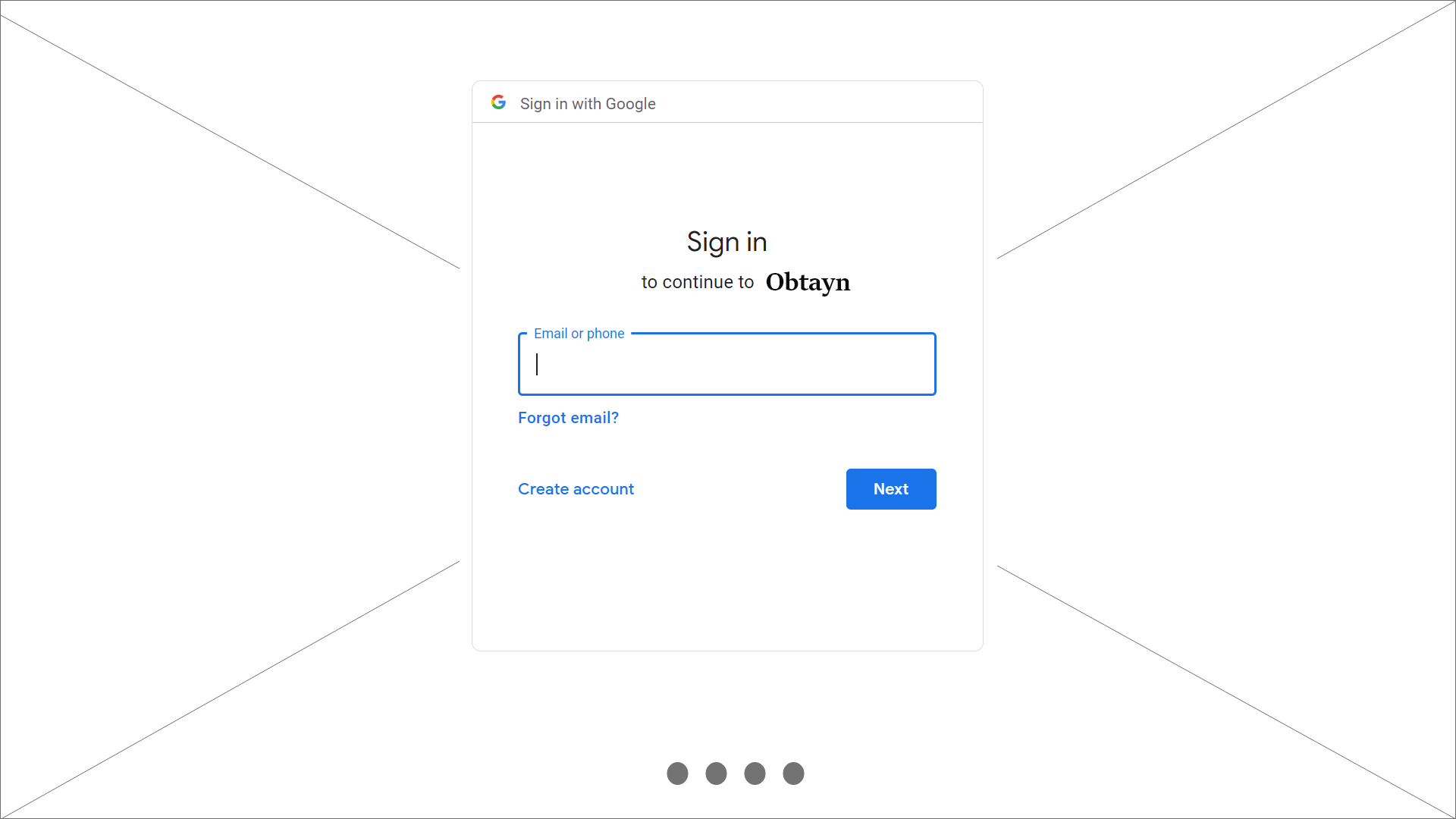
### Landing Page - Signup

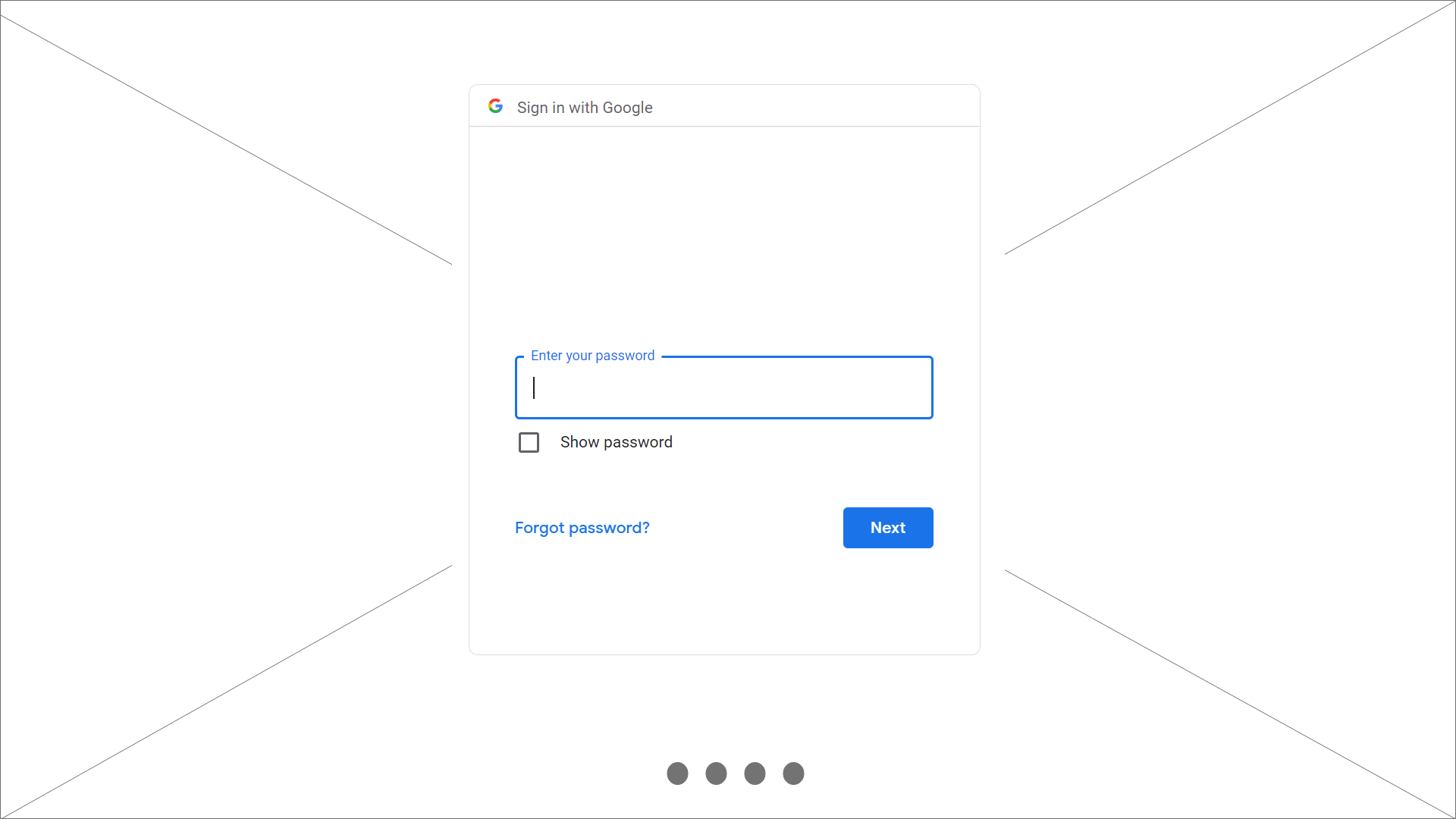
New users can register on this page. After successful registration, Users are redirected to Login.



### Continue with Google

Users can opt to use their Google accounts, for easy log in. Google asks them to input their email Id and password.

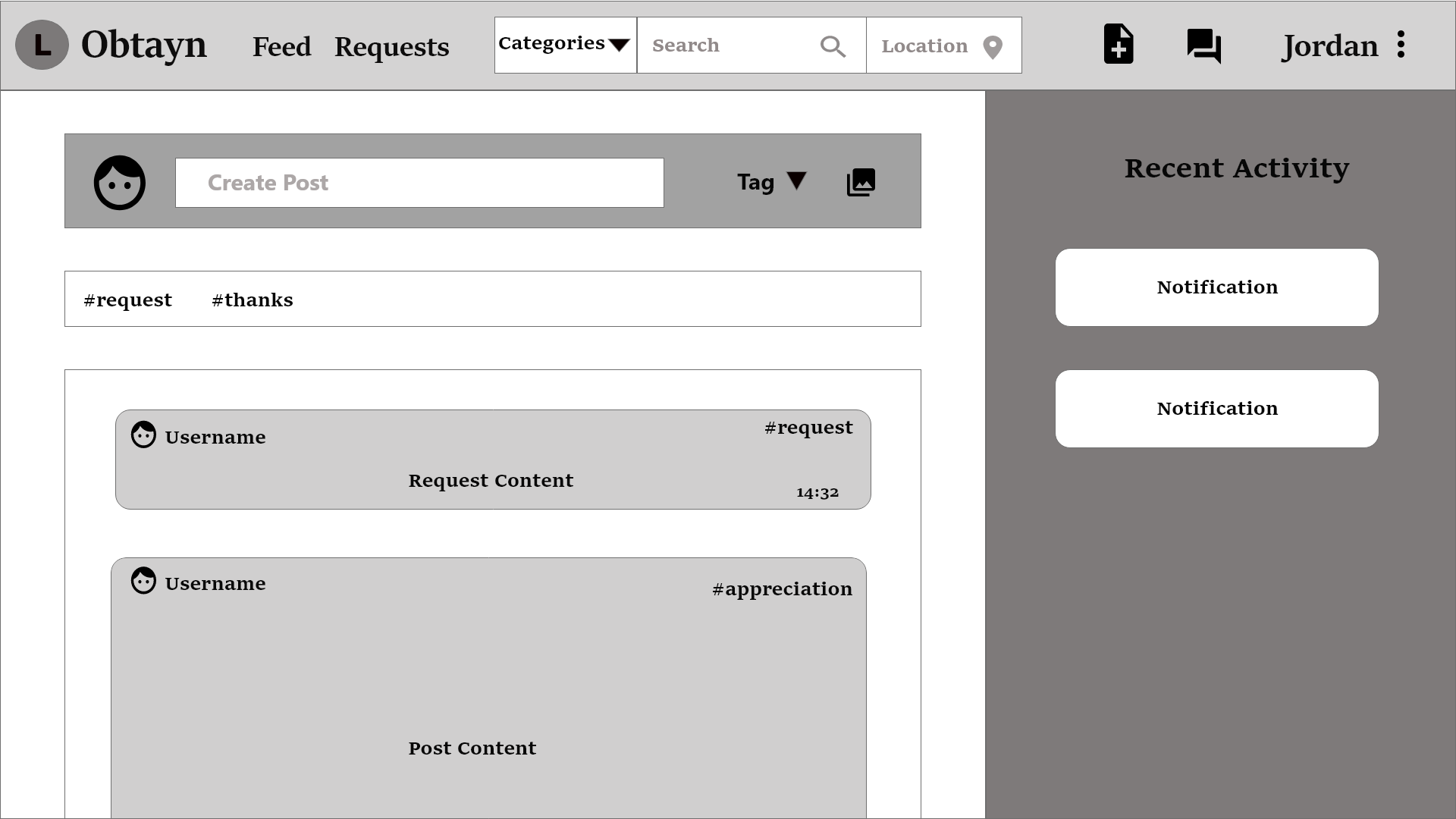




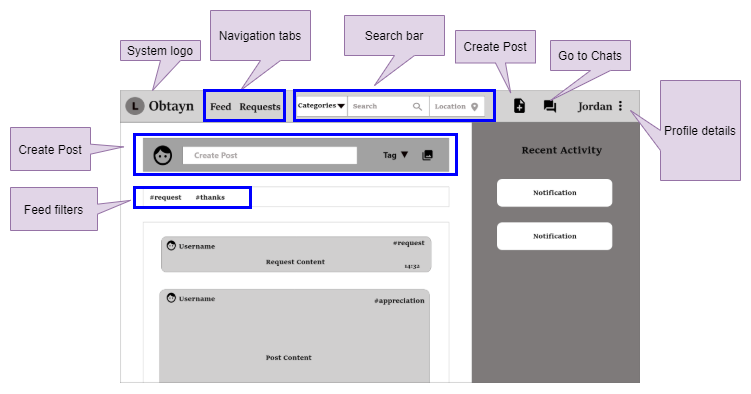
### Home Page

This page contains a list of posts which can be Requests or Appreciation posts. Users can also Create Post and view notifications. Search bar allows Users to filter the feed by categories, location, and also search for a particular post.

Users can navigate to the Requests page from the header.

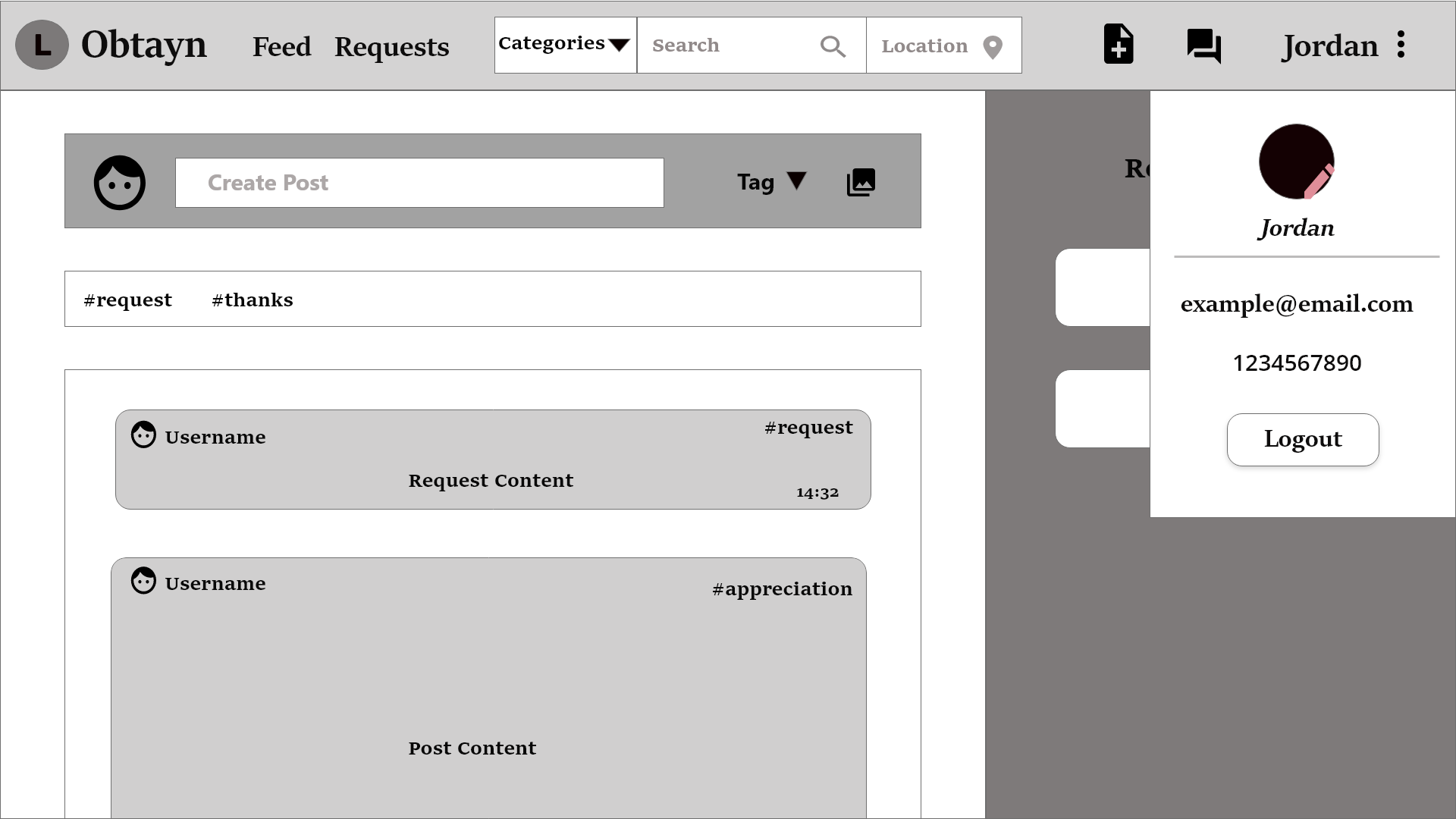


Below is a roadmap for the Homepage.



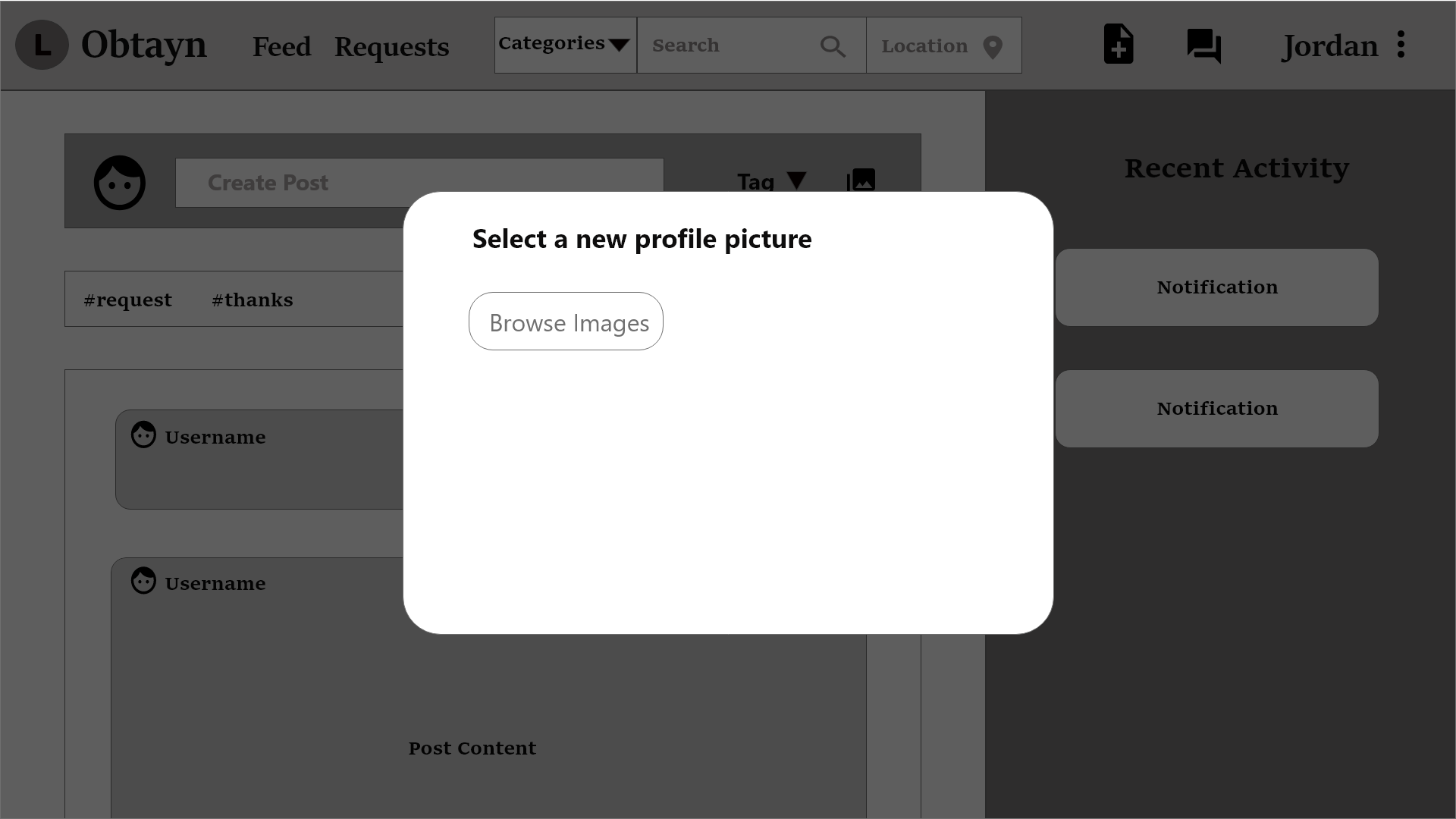
### View & Edit Profile Details

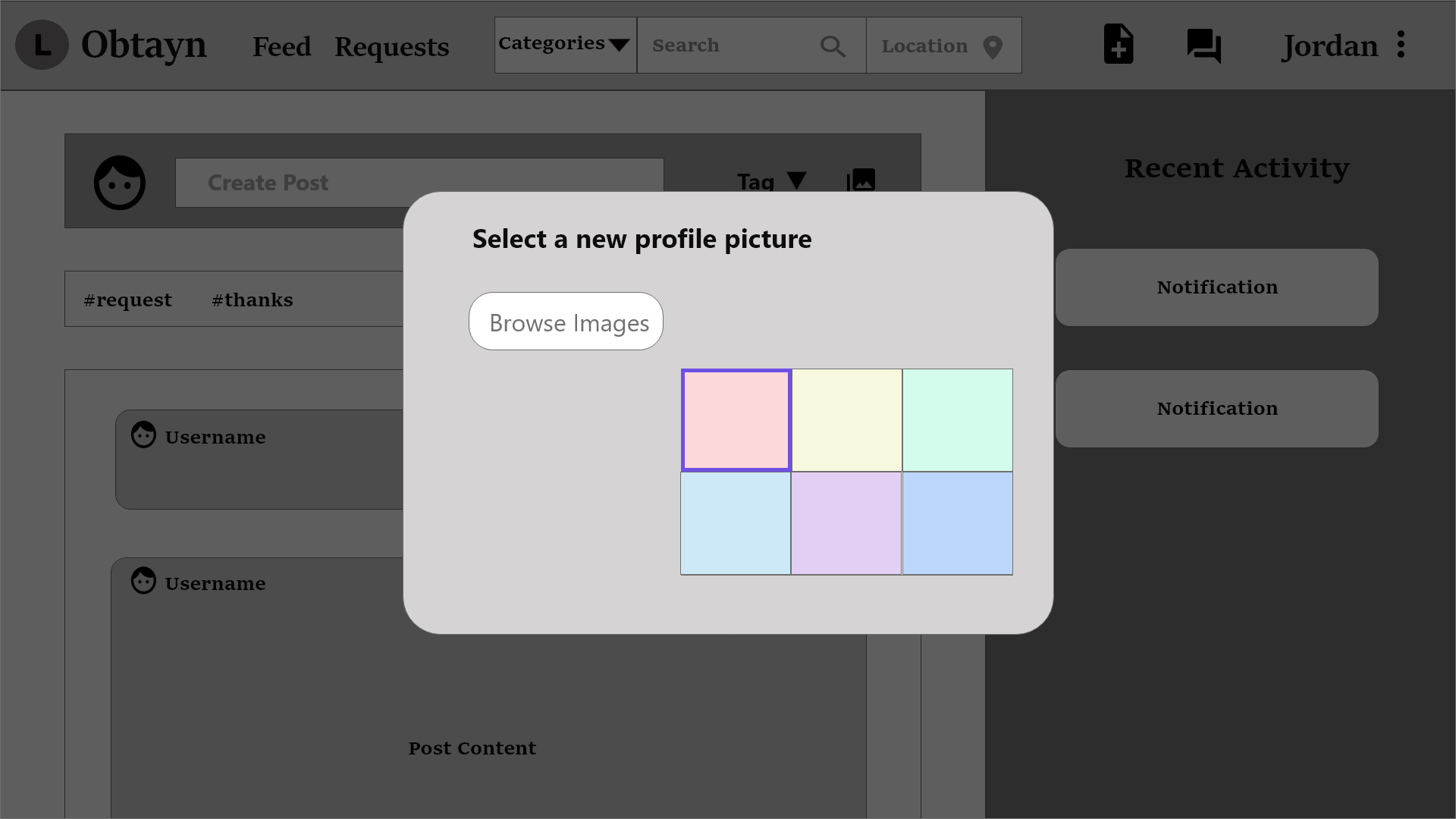
By clicking on their name in the header, Users can view their profile details, change profile picture and logout.



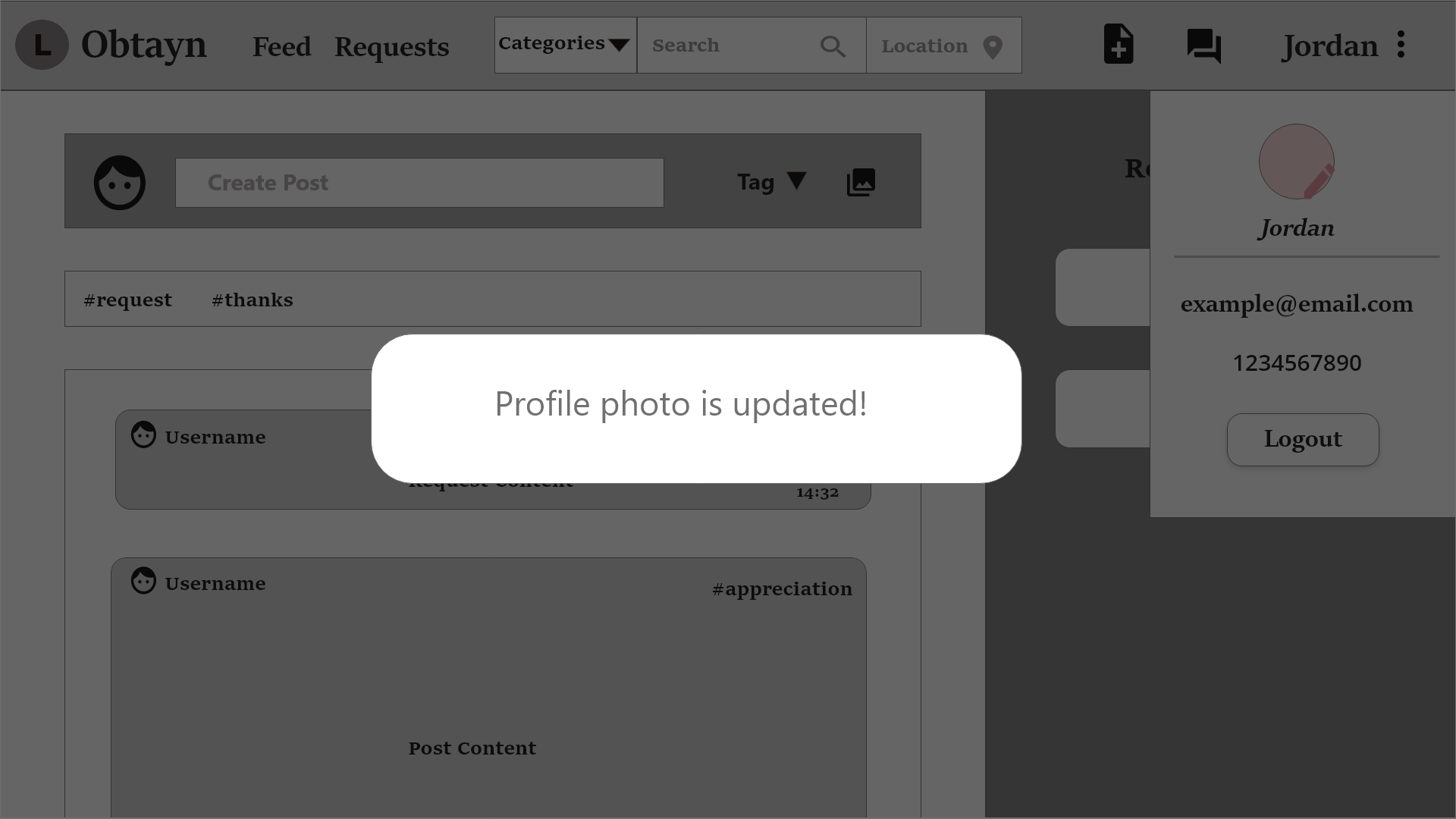
### Update Profile Picture

User prompts the user to browse images and select a new profile picture.



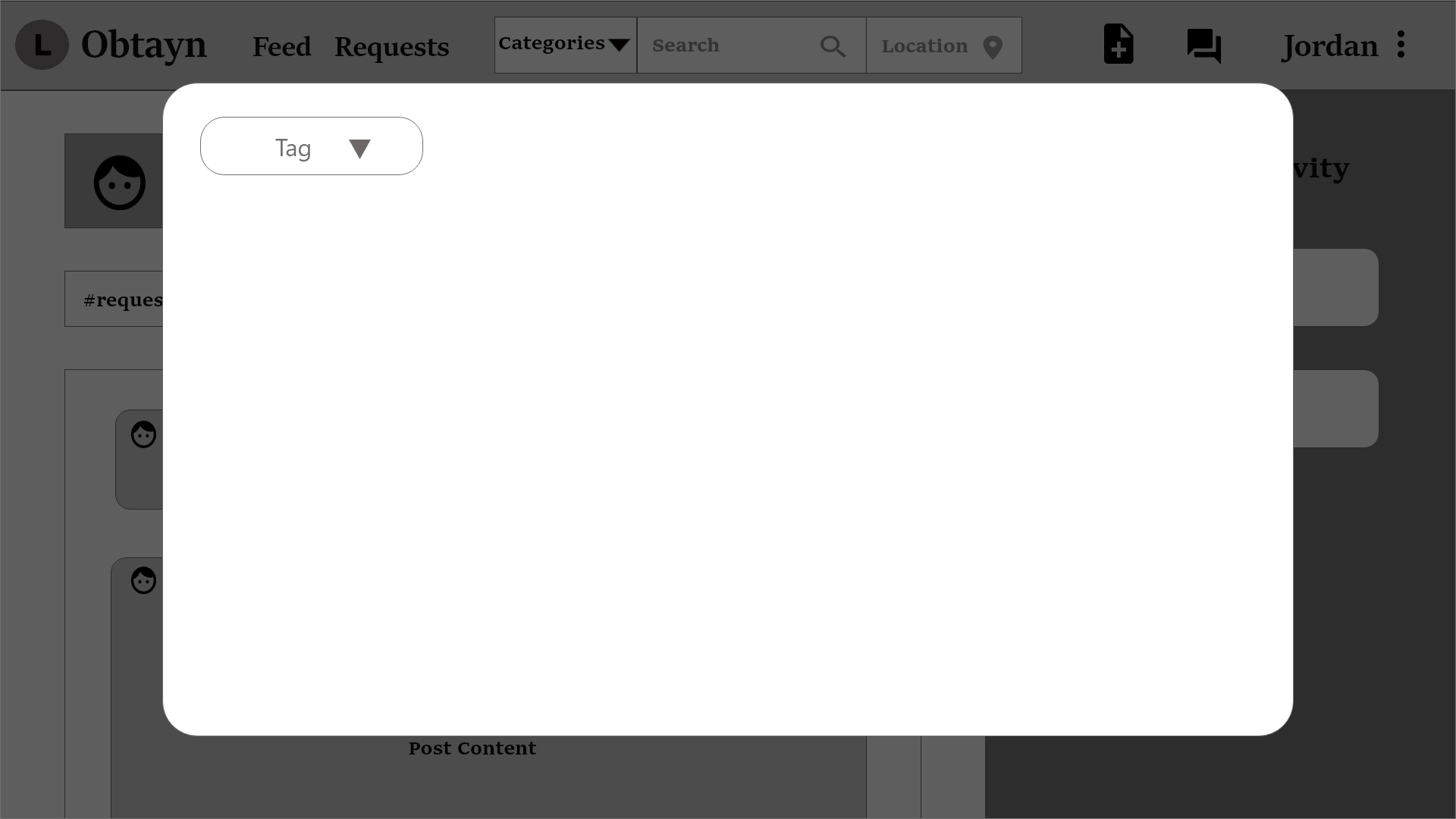


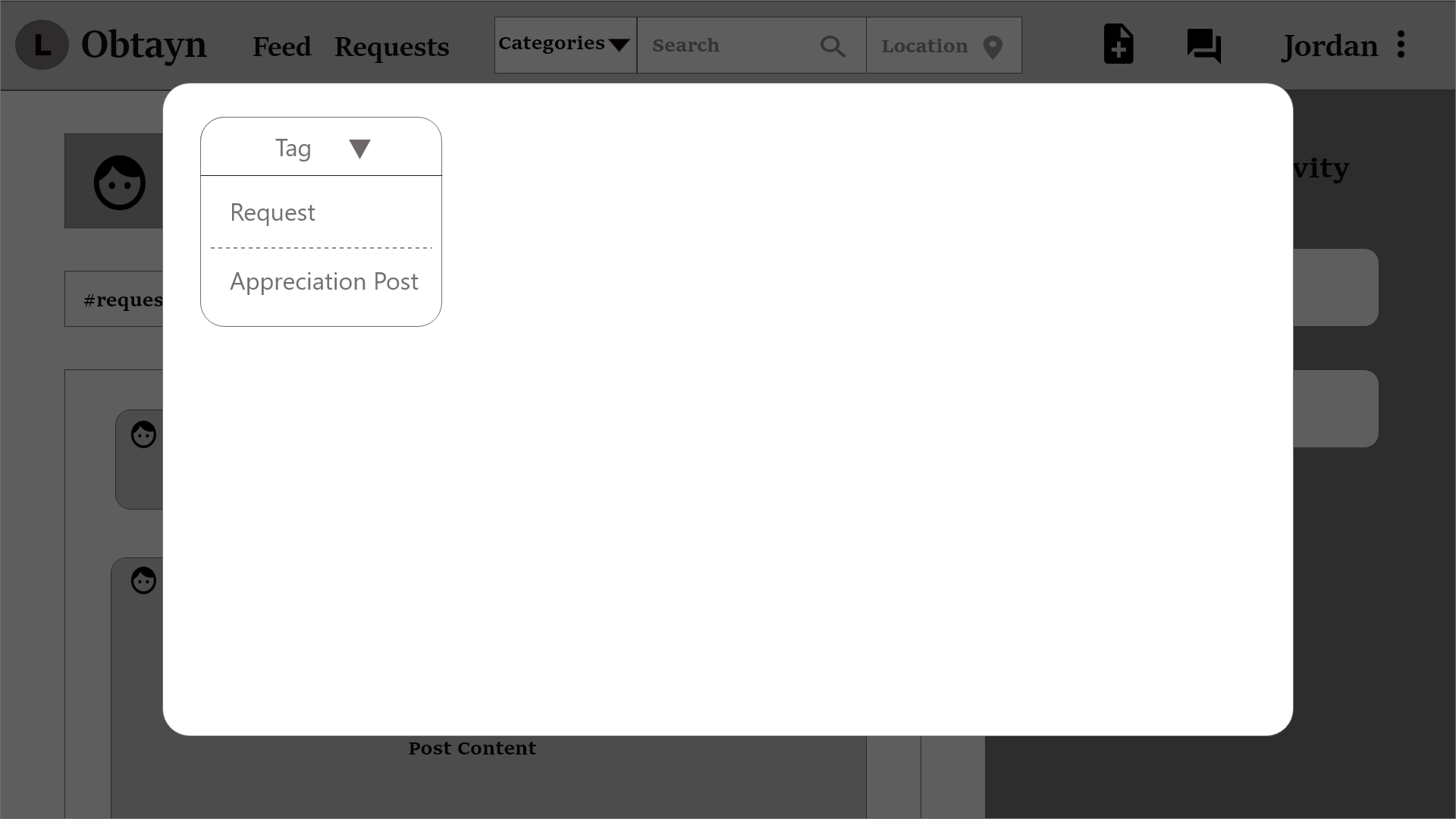
System displays a success message once the profile photo is updated successfully.



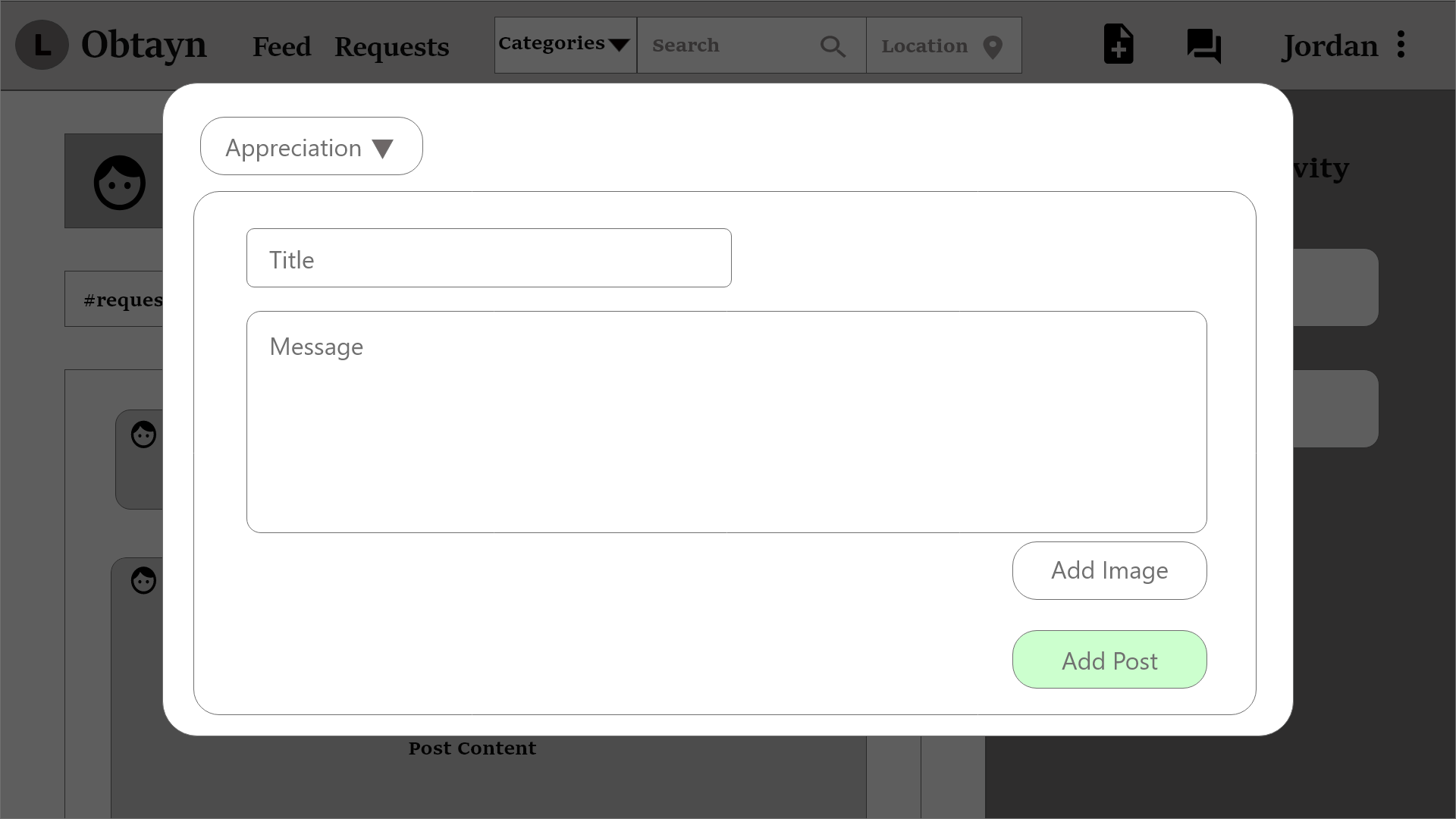
### Create Post

When Users click on Create Post in the header, or the one above the Feed, system opens a pop-up window. This window contains a dropdown list for “Tag” field, with 2 options - Request & Appreciation post. Tag describes the type of post.

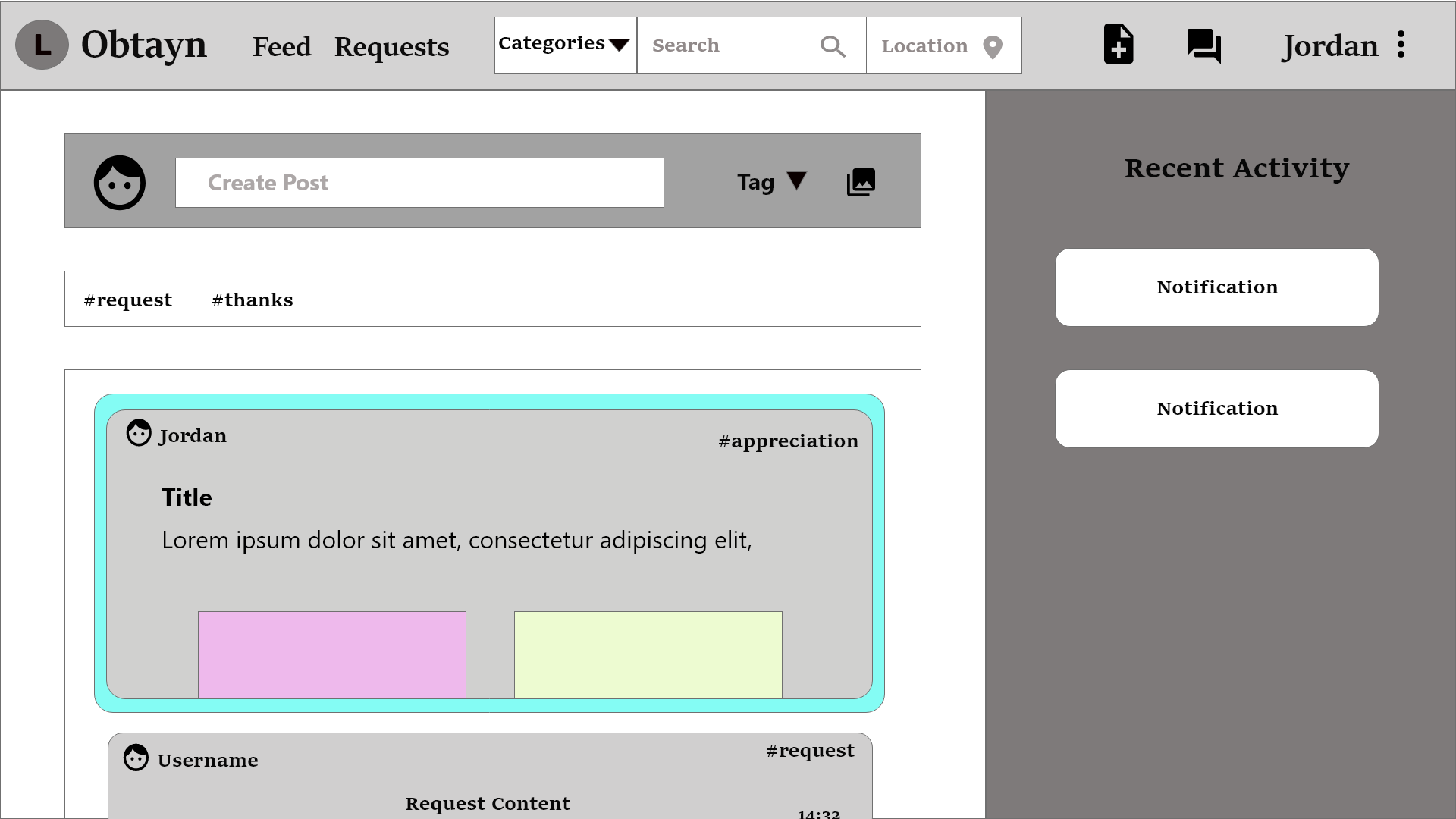


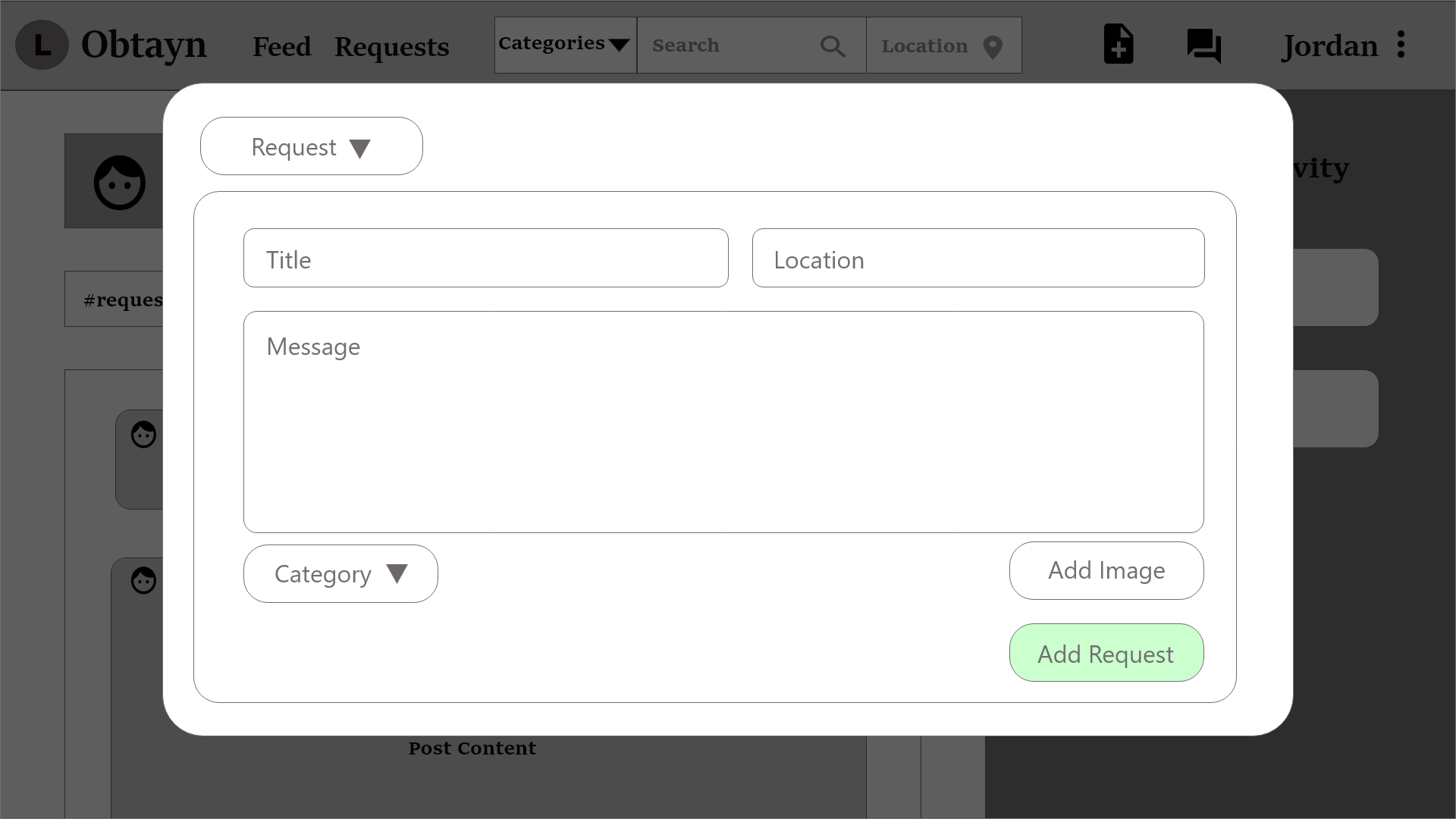


System displays form fields depending on the post Tag.

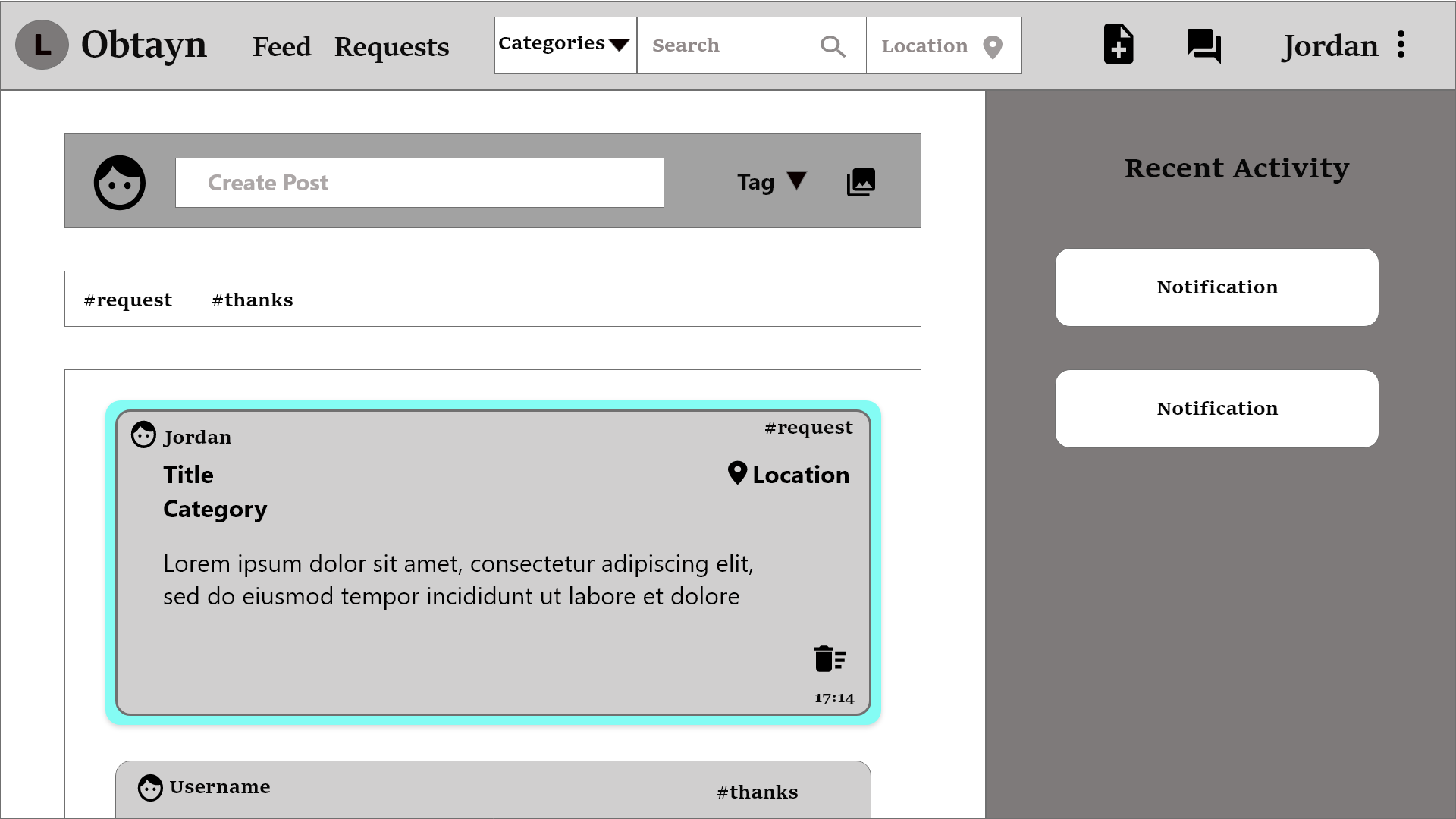


Once User clicks on “Add Post”, it shows on top of the Feed, and is highlighted.

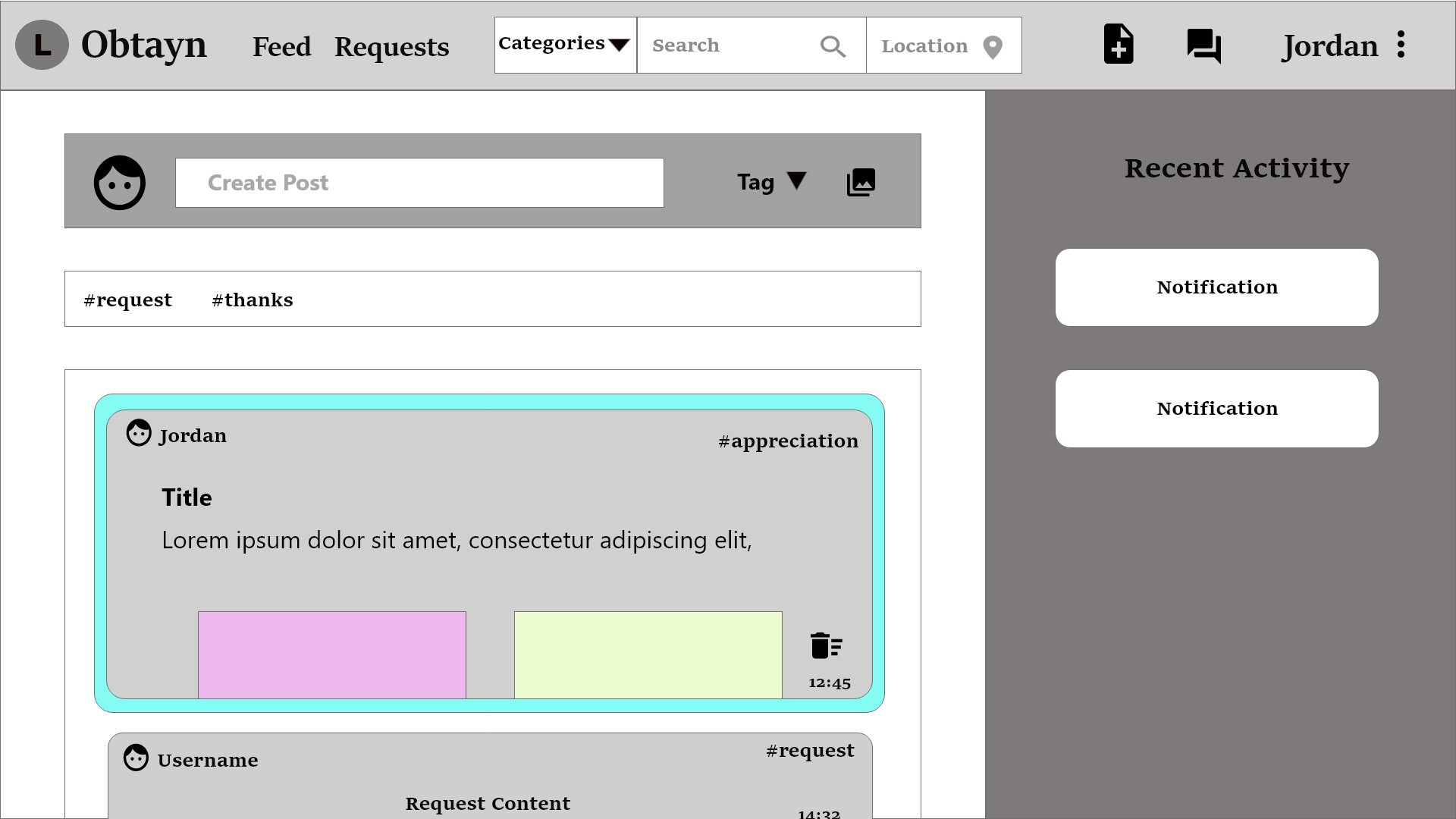




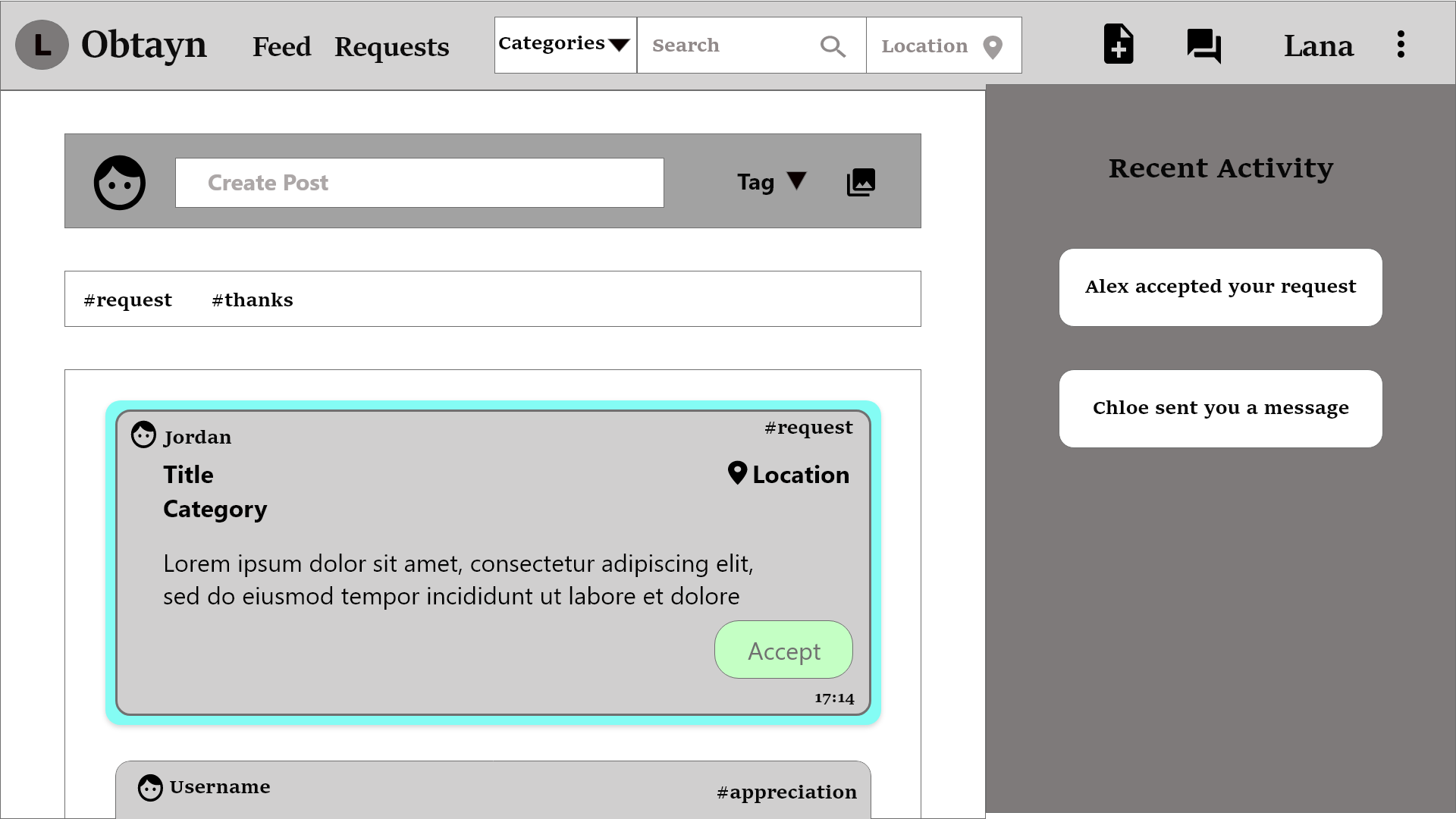
System disables User from accepting their own Request. Instead, Users can delete their Request by clicking the delete icon in the button-right corner of the post.



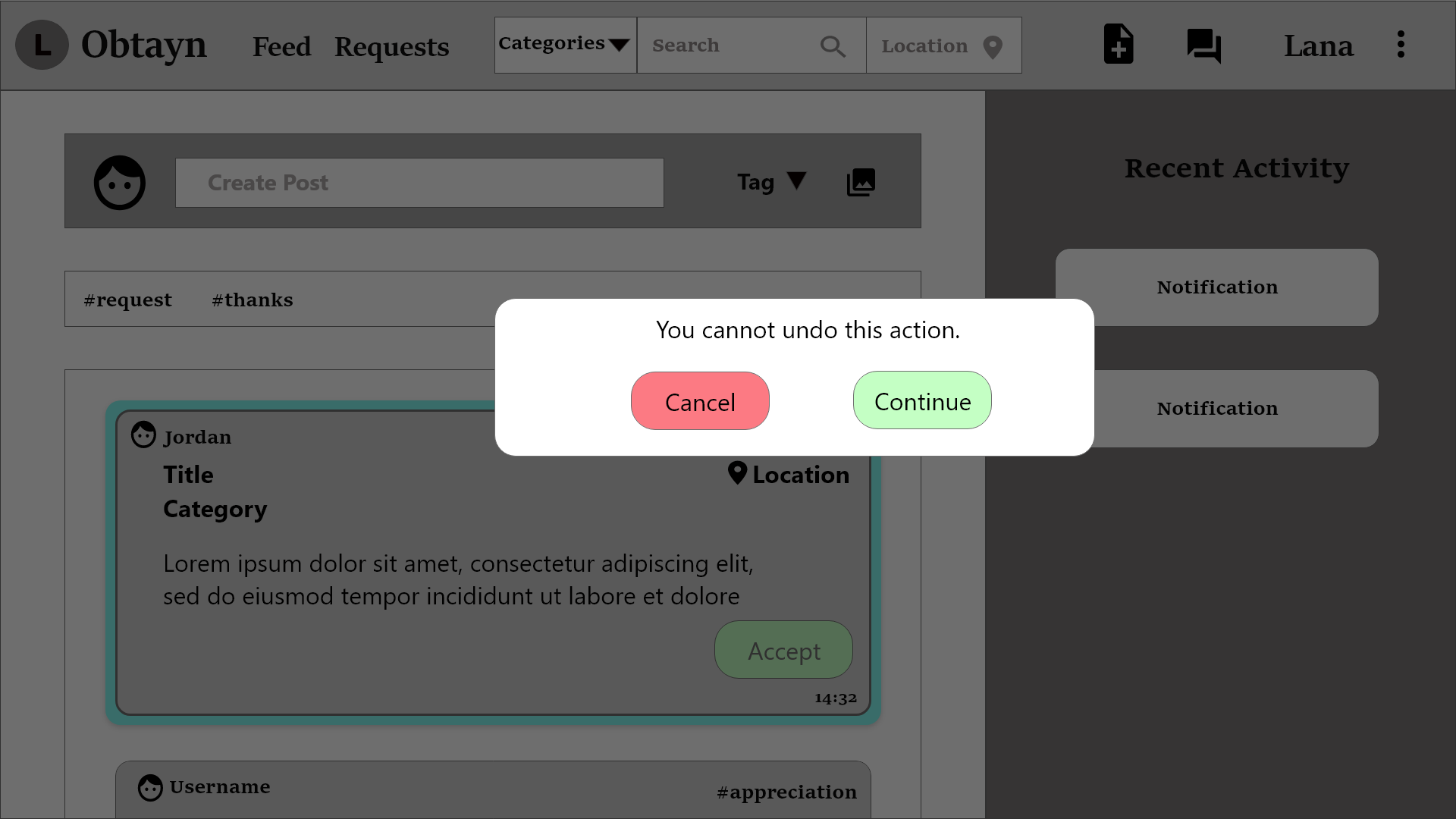
Similar to Requests, Users can delete their posts as well, by clicking the delete icon in the button-right corner of the post.



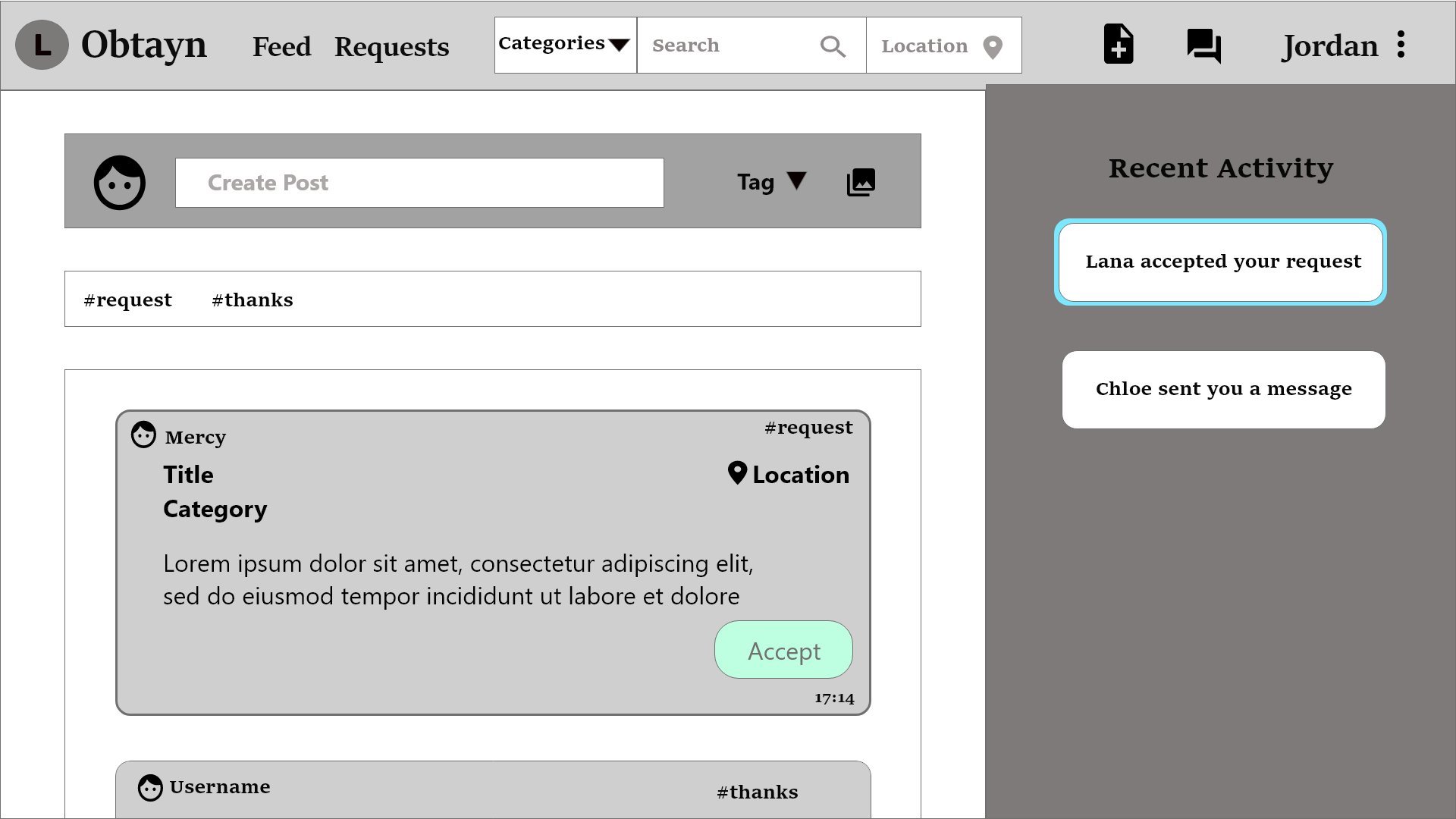
Other Users can accept Jordan’s request.



System displays a confirmation prompt when User clicks on “Accept”. If User clicks “Continue”, the corresponding request is removed from the Feed, and added to the User's Incoming Request.



Once Users’ request is accepted, they receive a notification.



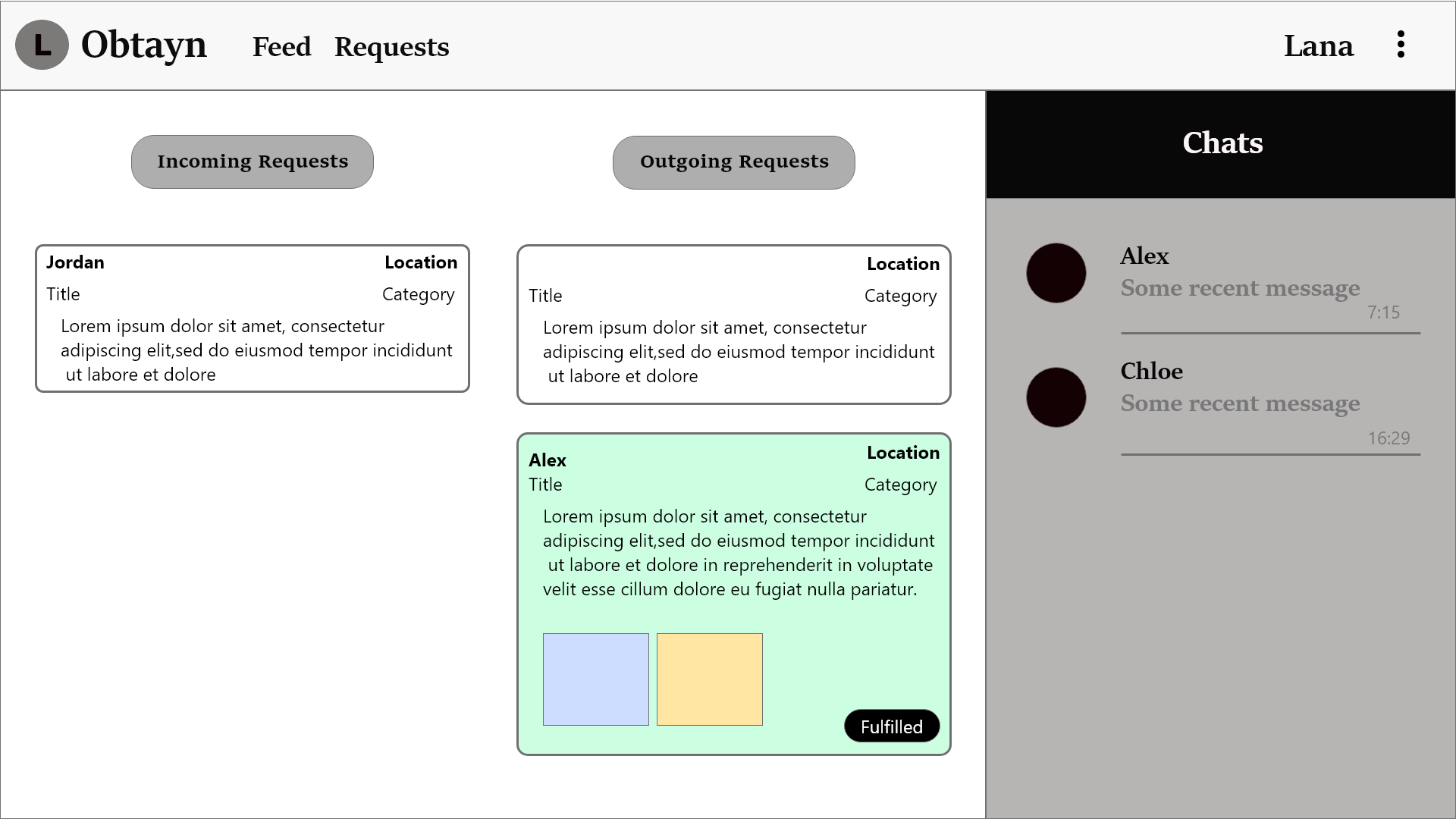
### Requests

Users can view Incoming and Outgoing requests from this page. They can also accept an incoming request.

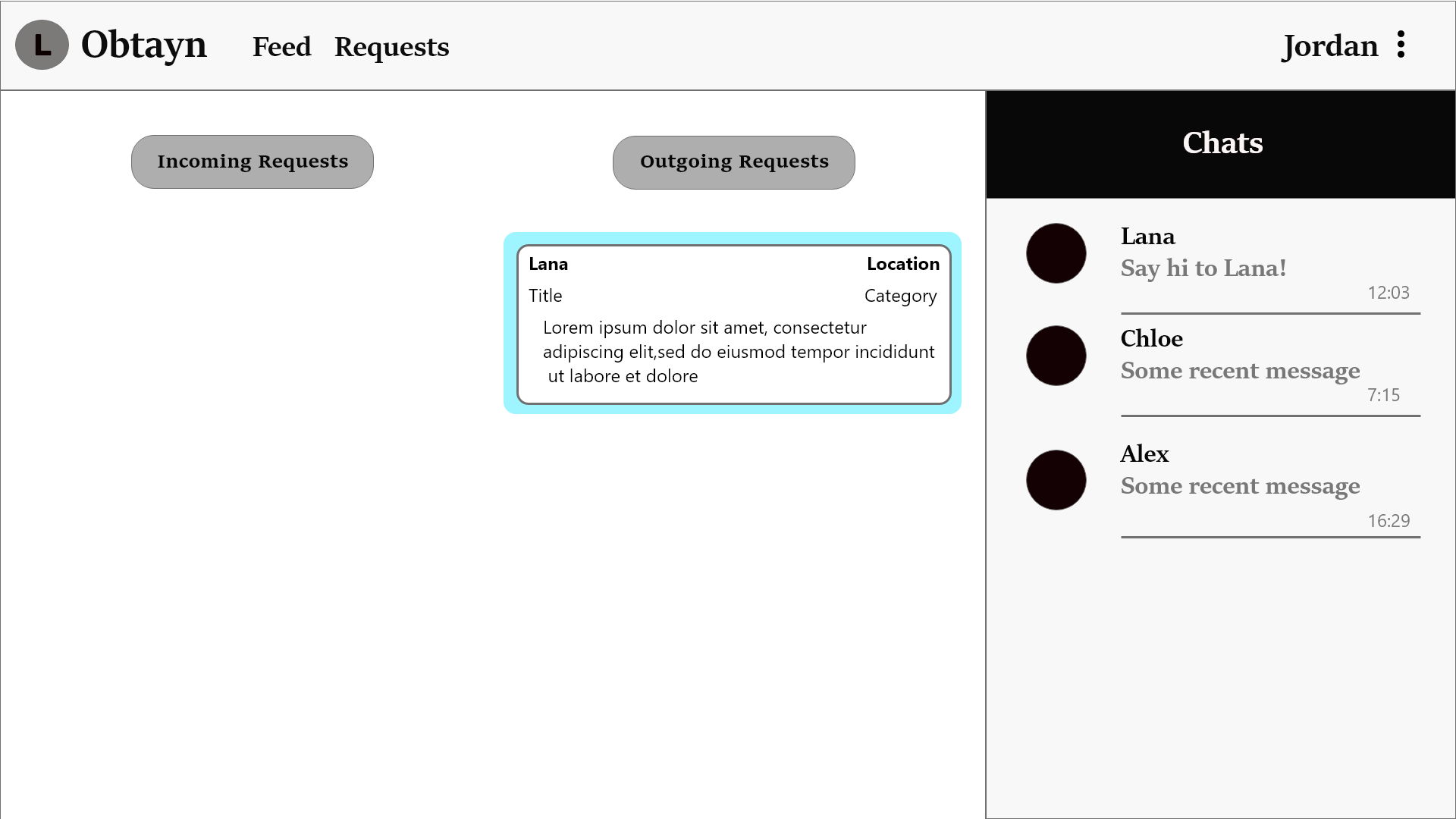
Incoming & Outgoing requests can be of 2 types - pending & accepted. Incoming requests highlighted in green denotes that request was accepted. Outgoing requests with a smiley are User’s requests that others have accepted.

Users can mark an Outgoing Request as Fulfilled, which denotes that (s)he has received his/her item.

The Chats section shows active chats. Users cannot chat until their request gets accepted.

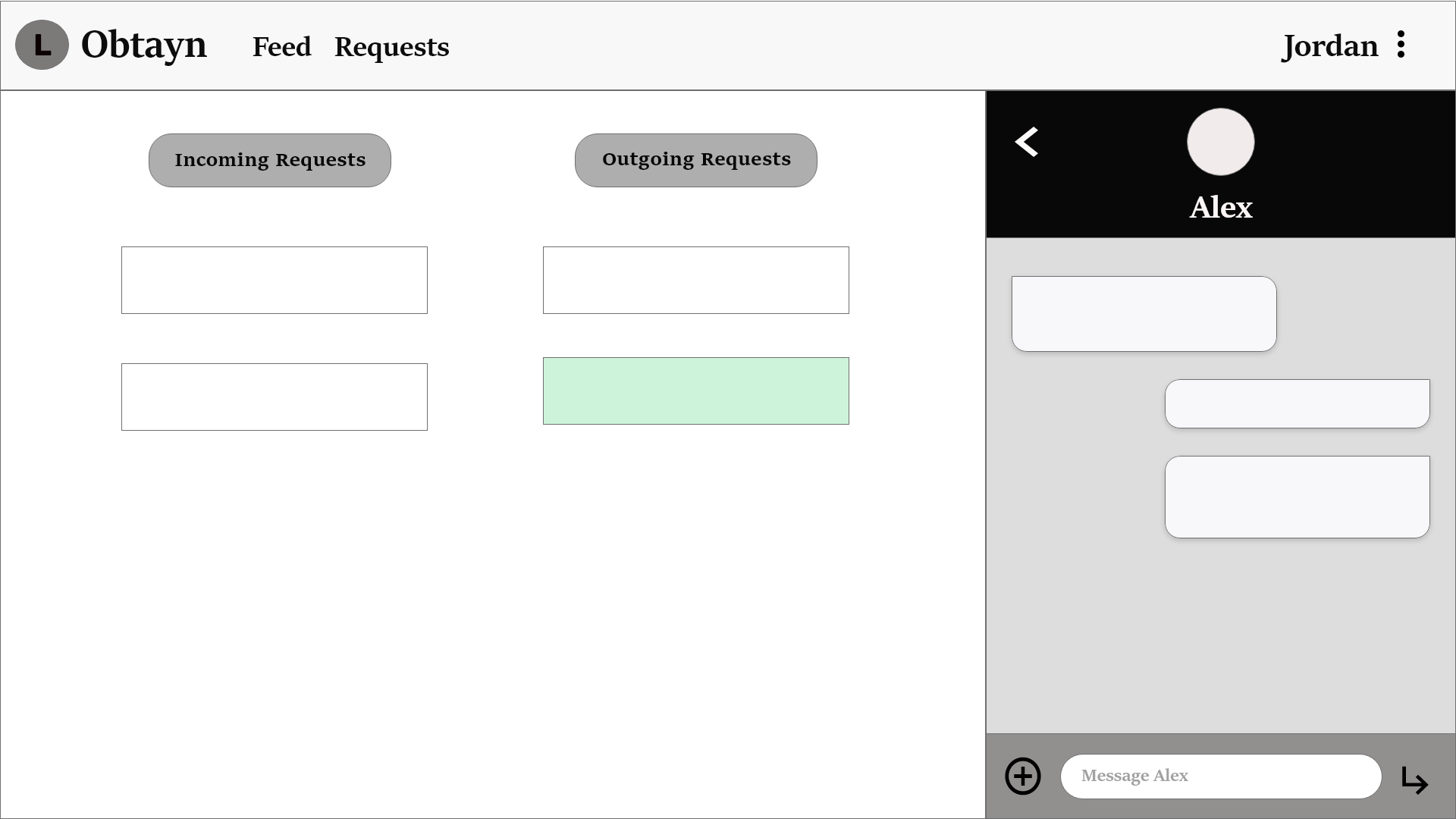


When a User’s Outgoing Request is accepted, (s)he can start a conversation with the request acceptor.



### Chat

Users can send text messages as well as images.



Users can also access Profile Details from the Requests page.

