Object-Oriented Analysis and Design: Project Part 2

Team: Neil Nistler, Kade Cooper, Brennan McConnell, Patrick Andresen

Title: Neighbor

Project Summary: Neighbor is a website designed to make the lives of its' users easier by providing a way to rent items to one another rather than buy them. The goal of this system is to crowdsource renting. We want to bring together a community which can provide desired items to each other for a short duration in which one user may have need of it while another user does not. In other words, the lack of such a system today forces users to only have the choice between eBay, Craigslist, or the shameless Facebook post to purchase an item. In actuality a person may only need said item for a short duration, or another person is unwilling to part permanently with his/her possession. An example would be, a user has need of a mountain bike for a weekend trip and posts to the website. Another user can then view all requests in his local area and can send an offer to the requester of the bike with the option to rent for $10 a day. An e-mail can be generated with the offer and be sent to the requester who can then accept or decline. (Optional to have a payment system in place, as well as a temporary security deposit). Ultimately, it would be a great way to borrow used items as well as earn some additional cash if you want to rent some of your items out. This system will be implemented as a website using Parse backend and the backbone.js framework. The high-level overview is a webapp that can handle many users and simultaneous requests. We are setting out to create the next modern form of purchasing items in today's day and age of crowd-sourced and interconnected lifestyles.

Project Requirements:

Business Requirements:

There are NO business requirements.

User Requirements:

ID	Requirements	Topic Area	Priority
US-01	As a User, I should be able to view my pertinent information at my homescreen. Pertinent information consists of the items I am renting from other people, the items I am renting to other people, and any offers/requests from another user about renting one of my available items.	Homescreen	Critical

US-02	As a User, I should be able to view other users' profiles/homescreens (which will include their personal bio and picture), but not the pertinent information that relates to them.	Others' Homescreen	High
US-03	As a User, I need to be able to search for available items to rent on the website.	Searching	Critical
US-04	As a User, I need to be able to click on an item in my search results list and see more information about the item. (An in-depth item info screen).	Searching	Critical
US-05	As a User, I need to be able to 'Make an Offer' or 'Accept the Renter's Offer' in order to show that I would like to rent the item from my fellow user.	Make an Offer	Critical
US-06	As a User, I need to be able to Sign-Up, and make a new account on Neighbor.	Sign-Up	Critical
US-07	As a User, I need to be able to Log-In to my existing Neighbor account.	Log-In	Critical
US-08	As a User, I need to be able to upload a profile photo as well as write a personal bio which will be displayed to others when they visit my homescreen.	Homescreen	High
US-09	As a User, I need to be able to post an item to Neighbor that I would like to rent out. I should be able to upload all necessary information, including photos.	Posting	Critical
US-10	As a User, I need to be able to accept/decline another users' offer on one of my available items. After we have exchanged money and the	Un-Posting	Critical

	item, clicking confirm request will remove the item from the search list.		
US-11	As a User, I need to be able to flag a post that is not in accordance with site guidelines to notify the owners that this posting should be removed.	Un-Posting	Medium

Non-Functional Requirements:

ID	**Requirements**	**Topic Area**	**Priority**
NFR-01	The system should be able to handle all user requests and connections without crashing.	Reliability And Performance	High
NFR-02	The database integrity must never be jeopardized and remain secure at all times.	Reliability	Critical
NFR-03	When a user sends an offer on an item, the alert on the owners' homescreen must appear within 1 minute and the e-mail that is generated must be sent within 1 minute.	Performance	High
NFR-04	The database can be maintained by the owners of Neighbor. And through direct interaction with the database interface Parse provides, we can alter/remove spam posts or posts not according to our site guidelines.	Supportability	High
NFR-05	When a user accepts/declines an offer on an item, the person who sent the offer must be notified within 1 minute.	Performance	High
NFR-06	Documentation must be provided that a user must read and accept when making an account with Neighbor. The documentation must	Usability	Medium

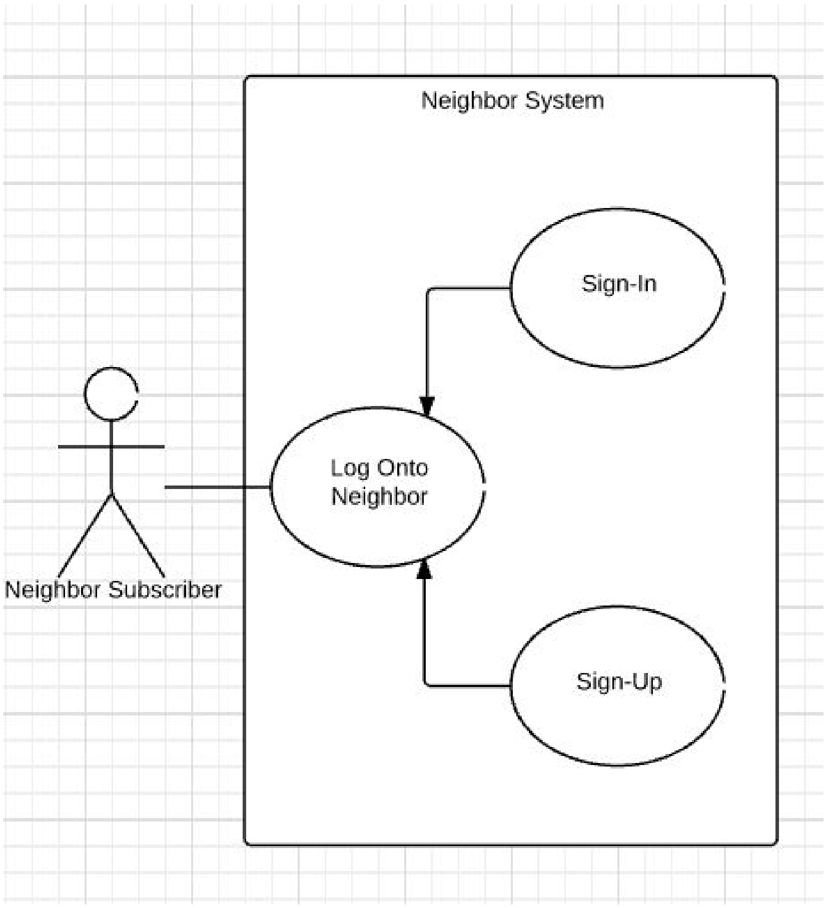
	include information on proper actions on Neighbor and what posts are not acceptable. (i.e. illegal items)		
NFR-07	Uploading pictures and submission of a new post to the website should take no longer than 10 seconds.	Performance	Critical

Functional Requirements:

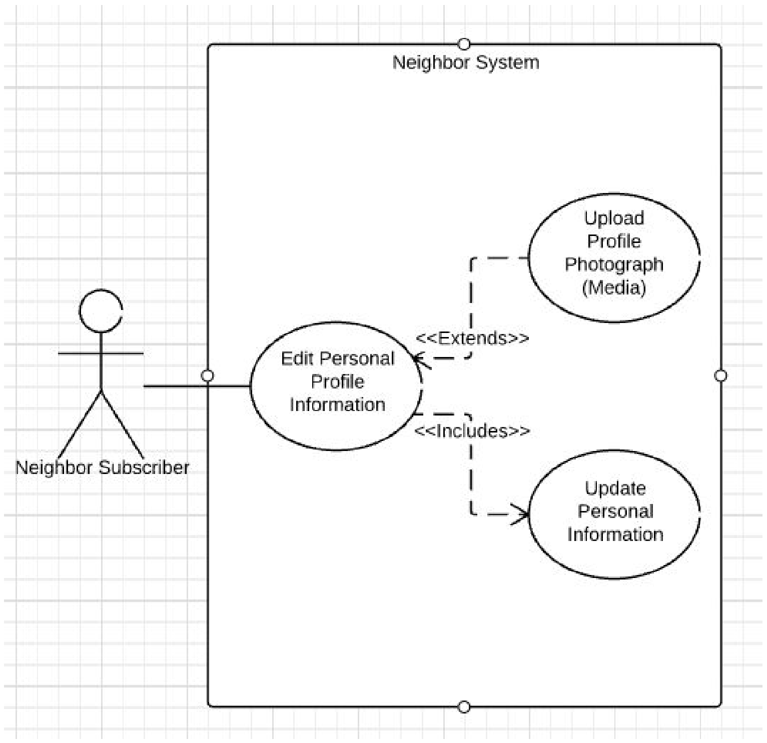
ID	**Requirements**	**Topic Area**	**Priority**
FR-01	The system will interact with MongoDB database via Parse.	Interface	Critical
FR-02	A maximum of 5 pictures may be posted for 1 item in a new posting.	Implementation	High
FR-03	A maximum of 1000 characters may be used to describe a new posting.	Implementation	High
FR-04	A personal bio may only contain 1 photo and at most 300 characters describing yourself.	Implementation	High
FR-05	A user must only be able to rent out the same item at 1 time. In other words, when the item is confirmed to be rented, any existing offers will vanish thus notifying the users who sent the offers.	Implementation	Critical
FR-06	The system will be managed by the owners of Neighbor	Operation	High

Users and Tasks:

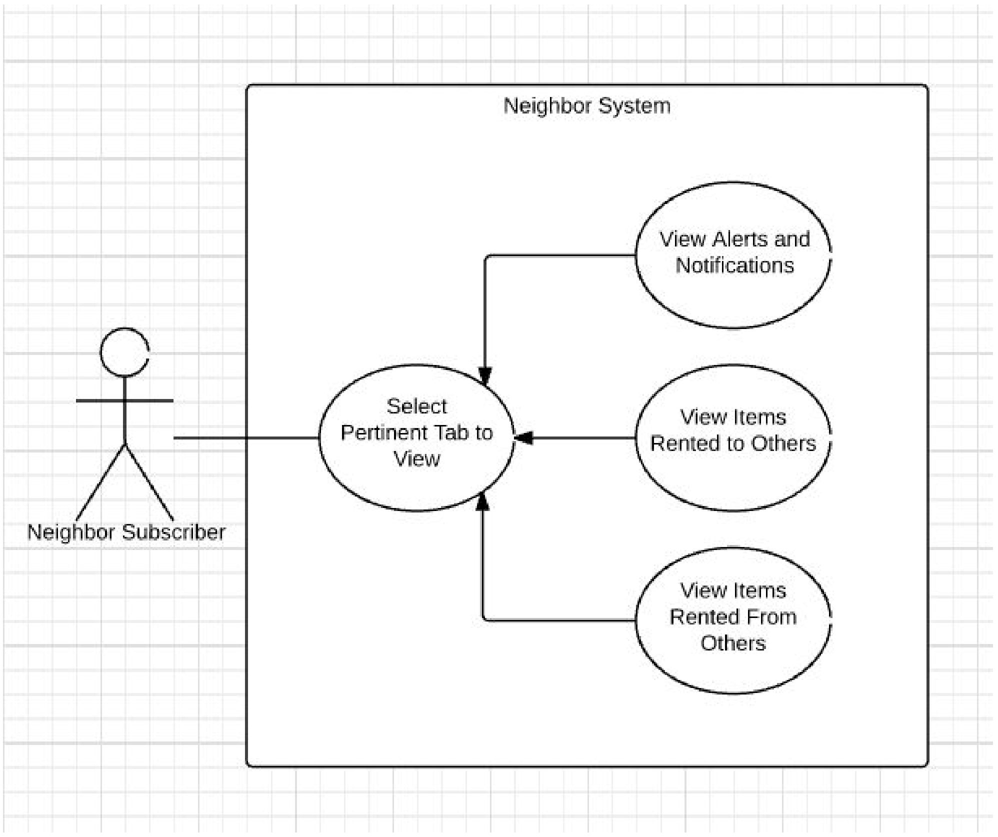
Use Case ID:	**UC-01**
Use Case Name:	**Log onto Neighbor**
Description:	**Users will either Sign-In to access their homepage, or be asked to Sign-Up in order to create a new account which they will then Sign-In to.**
Actors:	**Neighbor User**
Pre-conditions:	**The only pre-condition relates to the generalization for Sign-In, which requires that the account being signed into already has been created and exists.**
Post-conditions:	**The user will be redirected to their own home screen after logging onto the system.**
Frequency of Use:	**Daily by many different Neighbor Users**
Flow of Events:	**Actor Action : System Response** **1.) Navigate to Neighbor Website : Display index.html to User** **2.) Click either Sign-up or Sign-in : Display proper form** **3.) Provide information and enter : Verify Information. If no account found for a sign-in, redirect to the sign-up form. If information is valid, log the user into their personal home page.**
Variations:	**None**


Neighbor System
Sign-In
Log Onto Neighbor
Neighbor Subscriber
Sign-Up

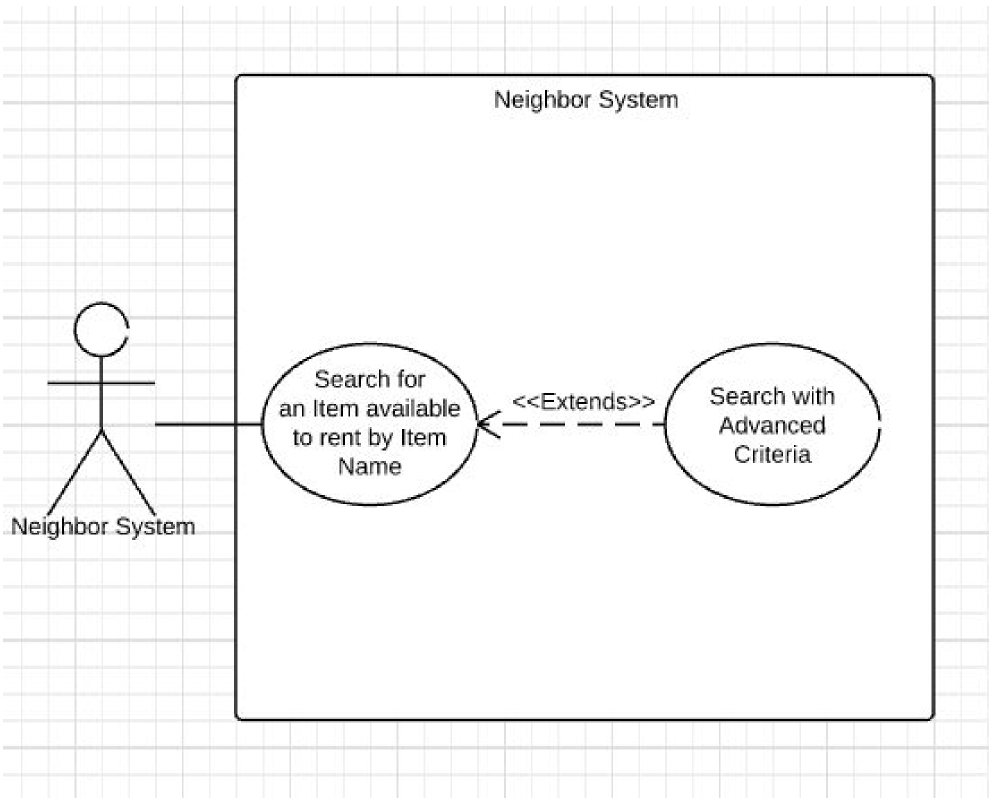
Use Case ID:	**UC-02**
Use Case Name:	**Edit Personal Bio displayed to other Users**
Description:	**Users should be able to select edit on their homepage and provide information about themselves including a short paragraph and a picture.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system.**
Post-conditions:	**Users' personal information will be updated and now be displayed on their home page for all users to see.**
Frequency of Use:	**Once per user, on average.**
Flow of Events:	**Actor Action : System Response** **1.) Click 'Edit Bio' Button : Display form for user to place info.** **2.) Click 'Upload Photo' Button : Display computer directory for user to select photo from.** **3.) Select photo from your files to use as headshot :** **4.) User types a short paragraph about themselves :** **5.) User clicks 'Submit Changes' : System updates the information in the database for that user, which will now be displayed on their homepage.**
Variations:	**User may only provide picture and not paragraph, or provide paragraph and not picture.**


Neighbor System
Upload Profile Photograph (Media)
<<Extends>>
Edit Personal Profile Information
<<Includes>>
Update Personal Information
Neighbor Subscriber

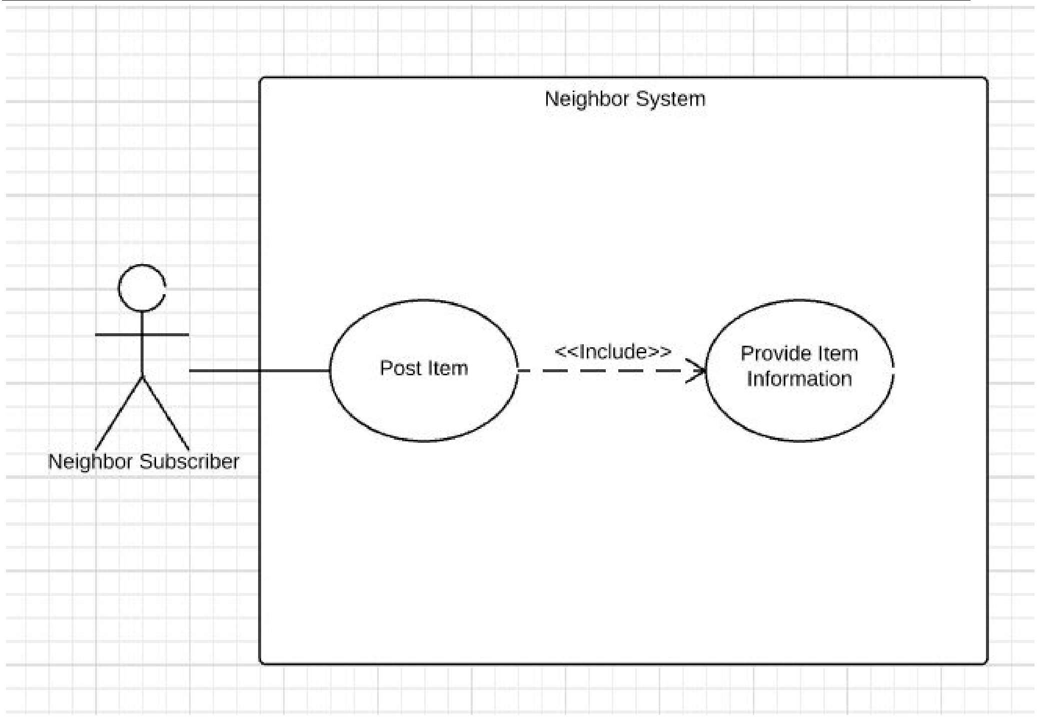
Use Case ID:	**UC-03**
Use Case Name:	**Select Pertinent Personal Information To View**
Description:	**Users will be able to navigate between three different tabs on their home page. 1st tab is items currently being rented TO others. 2nd tab is items currently being rented FROM others. 3rd tab holds user notifications such as offers from other users on one of their items or status of offers sent to other users.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and currently on their home page.**
Post-conditions:	**Depending on tab selected, relevant information will be updated and displayed on the screen for the user to view.**
Frequency of Use:	**Multiple times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) Click on the tab desired : Display the corresponding information retrieved from the database on the web page.**
Variations:	**None**


Neighbor System
View Alerts and Notifications
Select Pertinent Tab to View
View Items Rented to Others
Neighbor Subscriber
View Items Rented From Others

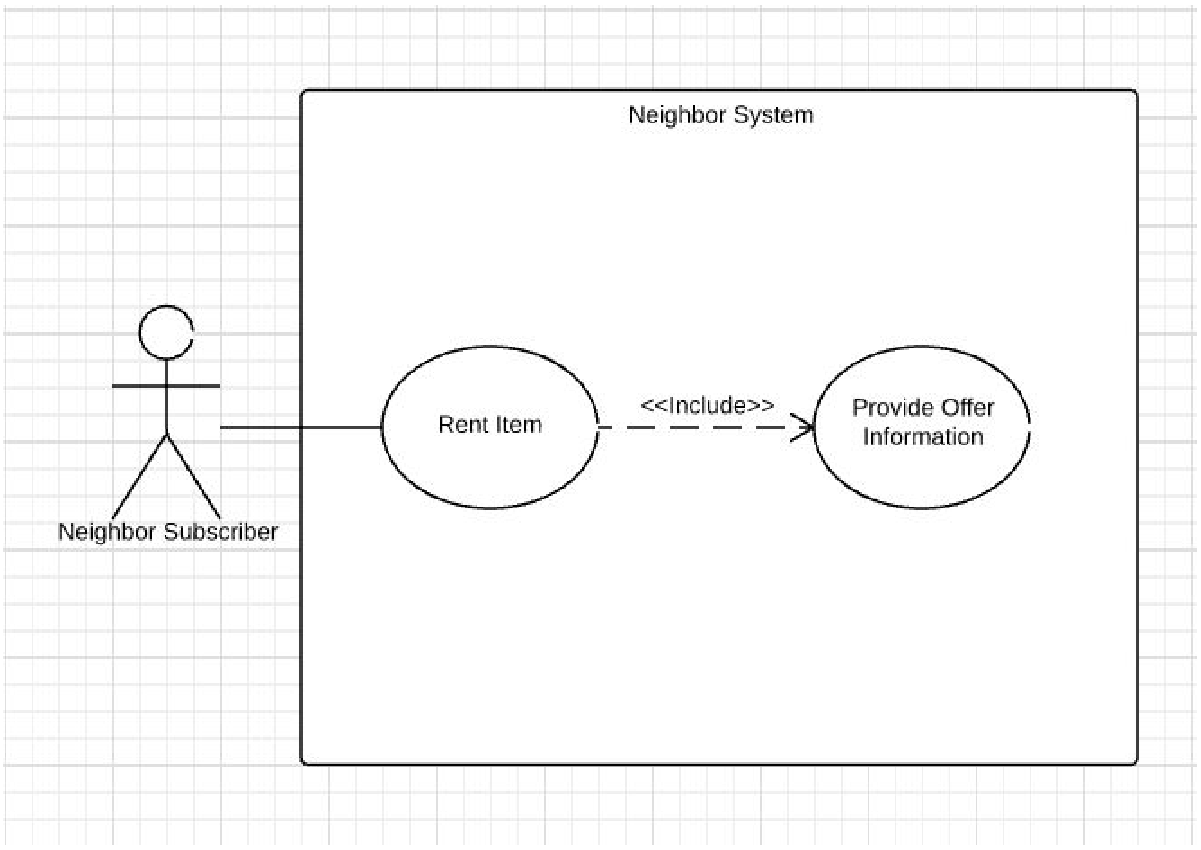
Use Case ID:	**UC-04**
Use Case Name:	**Search for an Item**
Description:	**Users will want to be able to search for certain items they are interested in renting, and view a list of all posts corresponding to their query.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system.**
Post-conditions:	**A list of all items the user may be interested in based off of their query will be displayed in a list format.**
Frequency of Use:	**Multiple times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) Enter in item desired and click 'Search' : Query the database for Items whose titles' contain the keyword and return a list of results which will then be displayed on the webpage to the user.**
Variations:	**None**


Neighbor System
Neighbor System
Search for an Item available to rent by Item Name
<<Extends>>
Search with Advanced Criteria

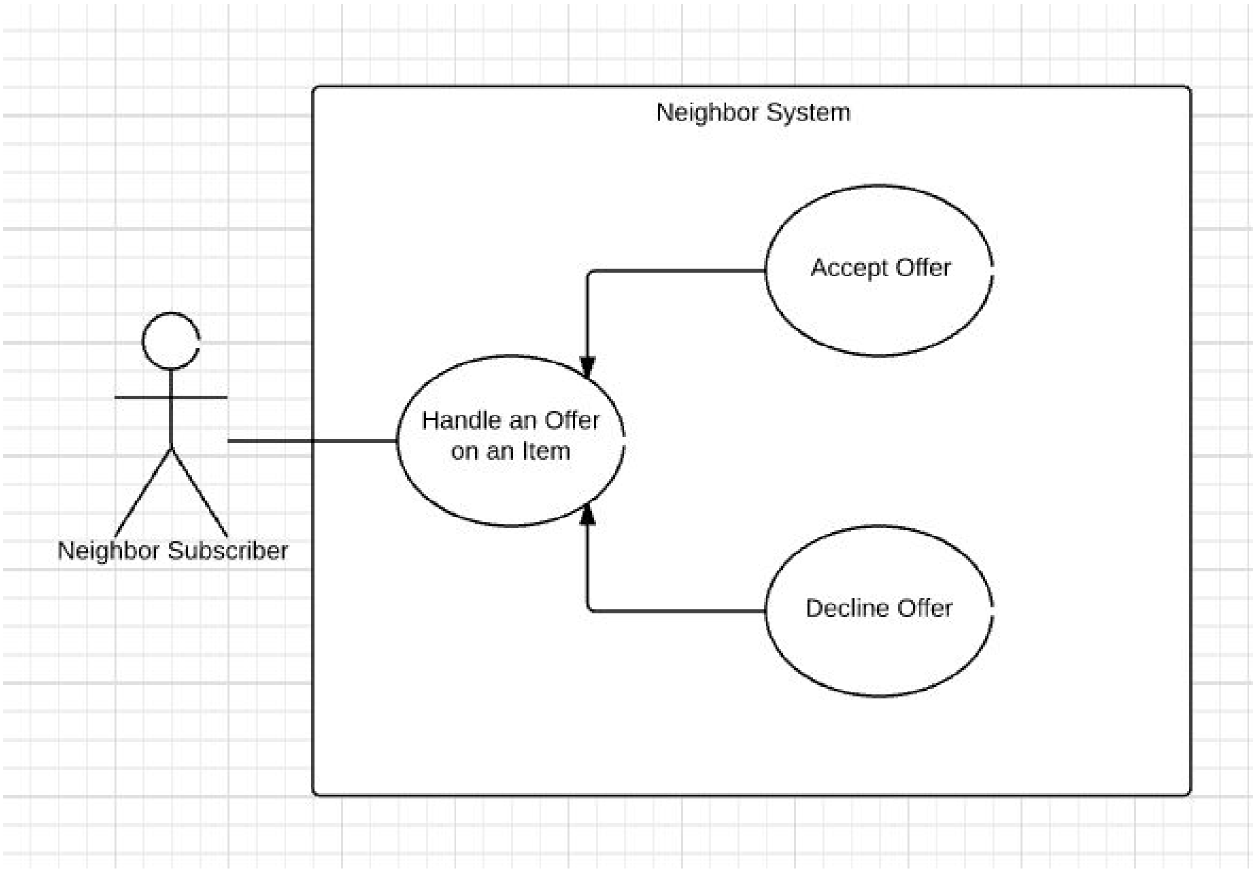
Use Case ID:	**UC-05**
Use Case Name:	**Post an Item**
Description:	**Users should be able to post a new item they are interested in renting out to others onto Neighbor. They will provide photos, a title, a description, an estimated cost per time rented, and other information in their post.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and currently on their home page.**
Post-conditions:	**After submitting their item post, it will be added to the archives and able to be displayed to others. User will be redirected to an in-depth view of their item just posted.**
Frequency of Use:	**Multiple times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) User clicks 'Post an Item' : System generates form object and displays it to the user** **2.) User provides necessary information including photos, title, description, and cost per time rented, and clicks 'Post'. : System closes the form and redirects the user to a view of the item they just posted.**
Variations:	**None**



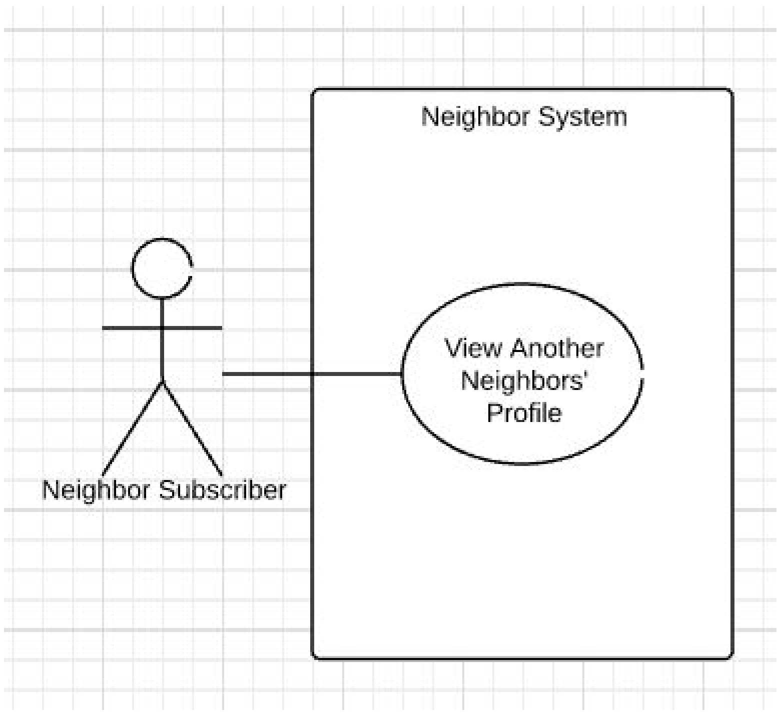
Use Case ID:	**UC-06**
Use Case Name:	**Rent an Item**
Description:	**Users should be able to send an offer on an item they would like to rent to the owner of the item.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and currently viewing the item they plan on renting.**
Post-conditions:	**After submitting the offer to the owner of the item they want to rent, a notification will be sent alongside an e-mail. The current user will be redirected to the query list.**
Frequency of Use:	**A few times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) User clicks 'Rent this Item' : A form object will be displayed** **2.) User will fill out the offer information on the form and submit : System will validate the form offer and send a notification to the owners homepage and send an e-mail to the owner.**
Variations:	**The user may choose to cancel their offer while filling out the form.**



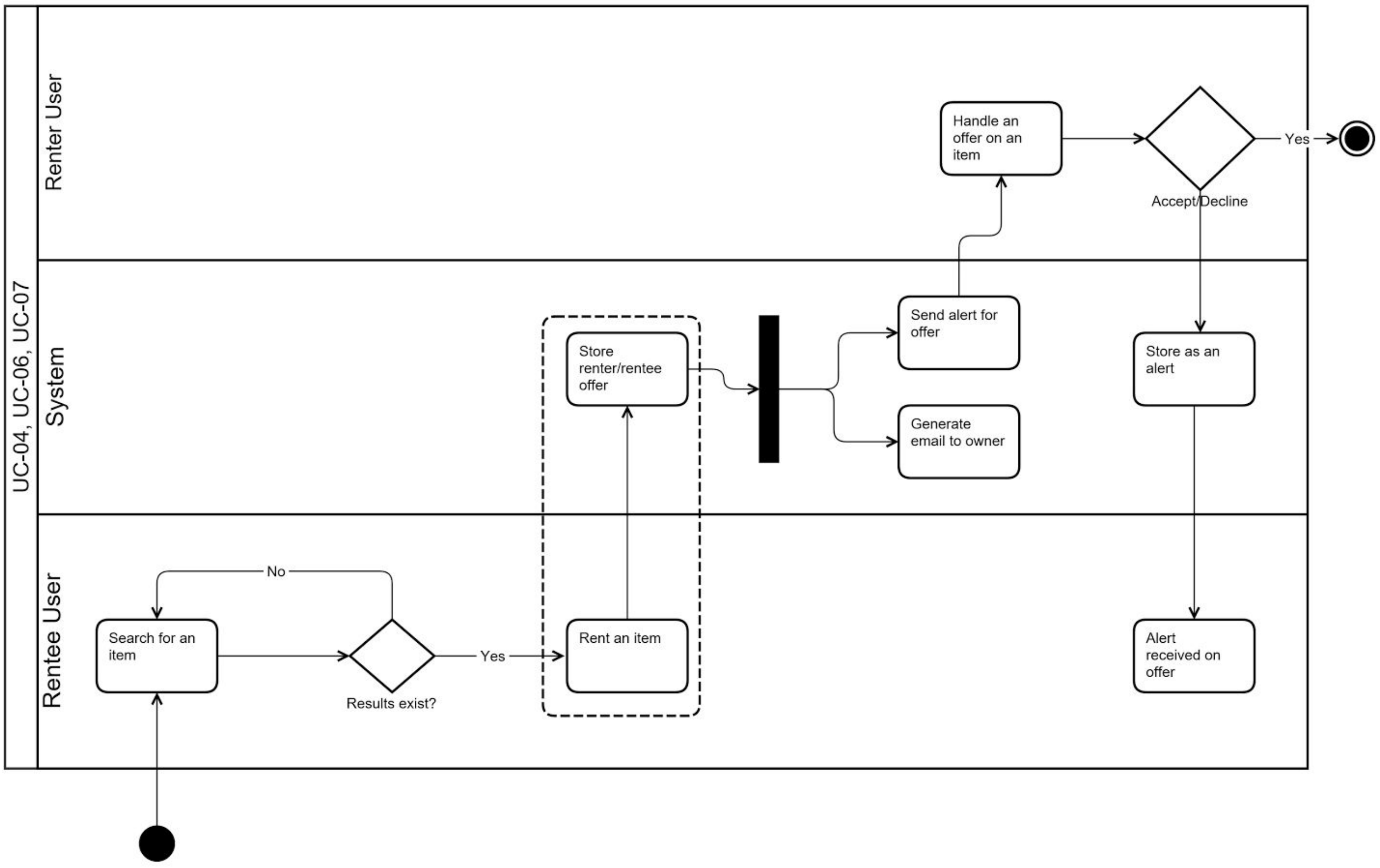
Use Case ID:	**UC-07**
Use Case Name:	**Handle an offer on an item.**
Description:	**Users should be able to accept or decline an offer on one of their available rental items.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and be on their homepage under the notifications tab.**
Post-conditions:	**Depending on the Users' action (accept or decline), a notification will be sent back to the user who sent the offer. If they accept the offer, their contact info will be sent back to the rentee.**
Frequency of Use:	**A few times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) User will see the notification in a list form, and be able to click 'Accept' or 'Decline' : The Renter's response will be sent back as an alert to the Rentee.**
Variations:	**None**



Use Case ID:	**UC-08**
Use Case Name:	**View other Users' Personal Bio**
Description:	**As a user, when viewing an item, I should be able to click on the owners name and get information about the owner of the item.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and be viewing an item.**
Post-conditions:	**After clicking on the owners name, the user will be redirected to the home page of the owner and be able to view their picture and personal paragraph info.**
Frequency of Use:	**A few times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) User clicks on another Neighbor users name : System will redirect the current user to the personal information of the user whom they clicked on.**
Variations:	**None**



Activity Diagram:



Data Storage:

<u>Requirements:</u>

Discuss how you will persist data in your application:

Since Parse Framework uses cloud-based technology, the main structure of the website along with each user will retain their data respectively and separately via subclasses within the *Parse.Object* framework. *Parse.Config* will also host the user's settings for each time they access the website.

What storage technology will you use? Text files? XML?sqlite?:

Parse Framework will store our data as JSON files. Parse uses MongoDB, a NoSQL database, as its backend and provides a useful interface to interact with MongoDB.

Where will the data be stored?

The data will be stored on the server side of our Parse Framework. Simple text objects/ fields will fit into *Parse.Object*(s) (i.e. a blob). Objects bigger than 10 MB such as images and videos will be stored in *Parse.File*(s).

Describe the classes that you will use to access this data at runtime. Talk about this

1. User - As a Parse.Object, User with subclasses (Subscribers) will either login with their credentials to view their Items/make changes to their own settings (Subscribers) or be able to create an account within the website (A Guest becoming a Subscriber)
2. Item - As a Parse.Object, Items will be created, edited, and or deleted by Users and can be attached with Multimedia Items (A subclass calling upon Parse.Objects). They will also be able to display appropriate information about themselves (Title, summary of the object, rentee, renter, etc.)
3. Screen - As a Parse.Object, window(s) will be able to populate the screen with subclasses of menu(s), button(s), scrollbar(s), and titlebar(s).

Links:

https://parse.com/ https://parse.com/docs/js/guide#objects

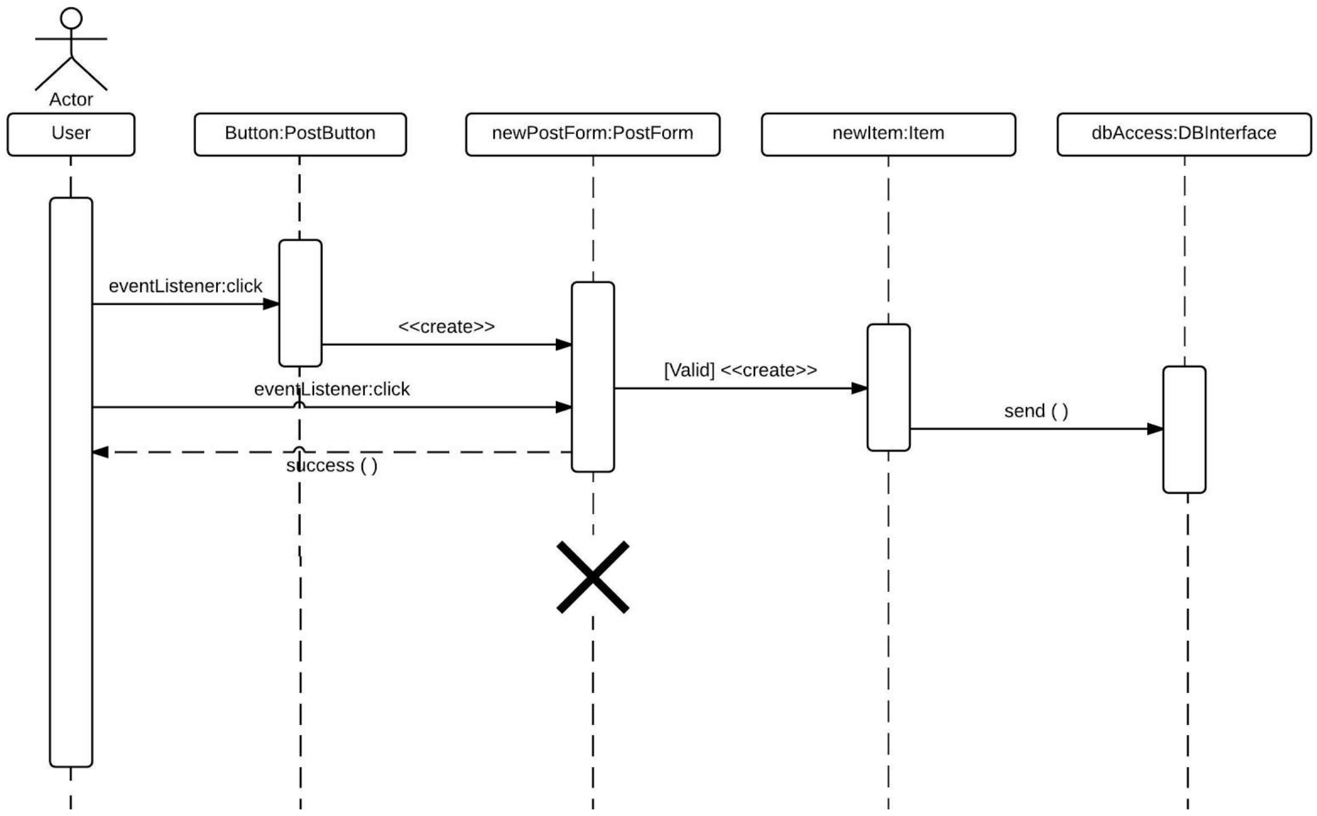
UI Mockups: (See End of Document for Mock-Up Diagrams)

A user will transition between 5 screens. The first screen a user will be presented with is the Log-In page in which a user will be able to enter their information to sign into their homescreen. If they do not have an account, they can say so on the Log-In page and they will be presented with the Sign-Up screen in which they can make an account with neighbor. On the homescreen, a user will have a profile photo, a personal bio, and a search bar to begin searching Neighbor for posts. The home screen also has 3 different views for the screen based on what tab is selected by the user. They can see the items they are currently renting out to others, items they are currently renting from others, as well as requests/notifications about users who want to rent one of their items which they can confirm/accept. There is also the search screen which displays relevant query results in a list format. And lastly there is an in-depth object view if a user clicks on an entry in the query results list. The user will see various links throughout their transitioning of the website such as links to view information about other Neighbors' profiles, as well as links to send an offer on an item which will generate an alert and send an e-mail to the owner of said item. We want the website to be simple and easy to navigate with useful information being displayed. We will avoid redundancy and emphasize pertinent information.

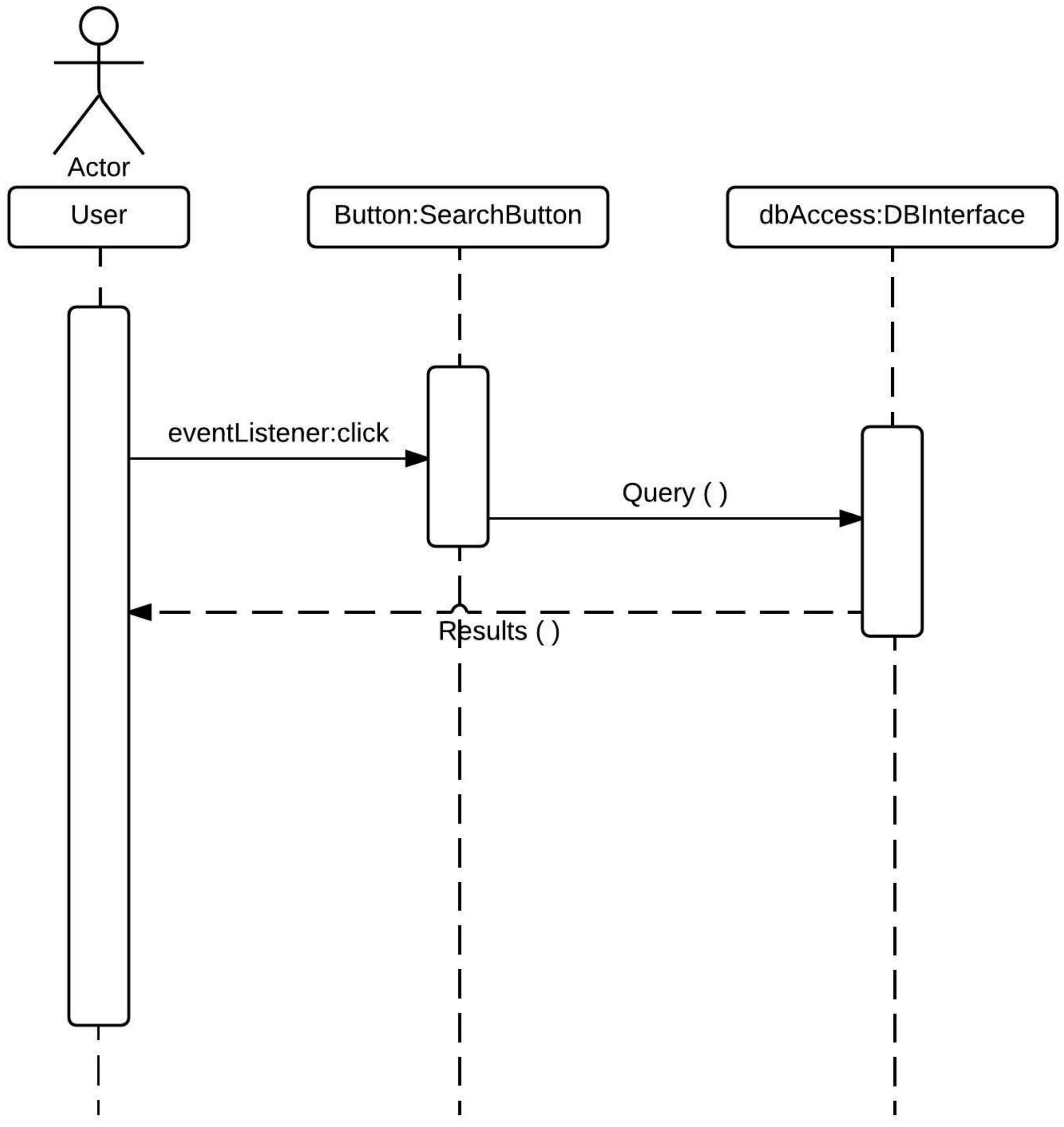
User Interactions:

While Neighbor has many user interactions, all of which are necessary, some are trivial and not worth elaborating on. However, other interactions, are quite convoluted and involve interacting with many parts of the system. We have chosen to model the most important and interesting actions that a user will make while using Neighbor.

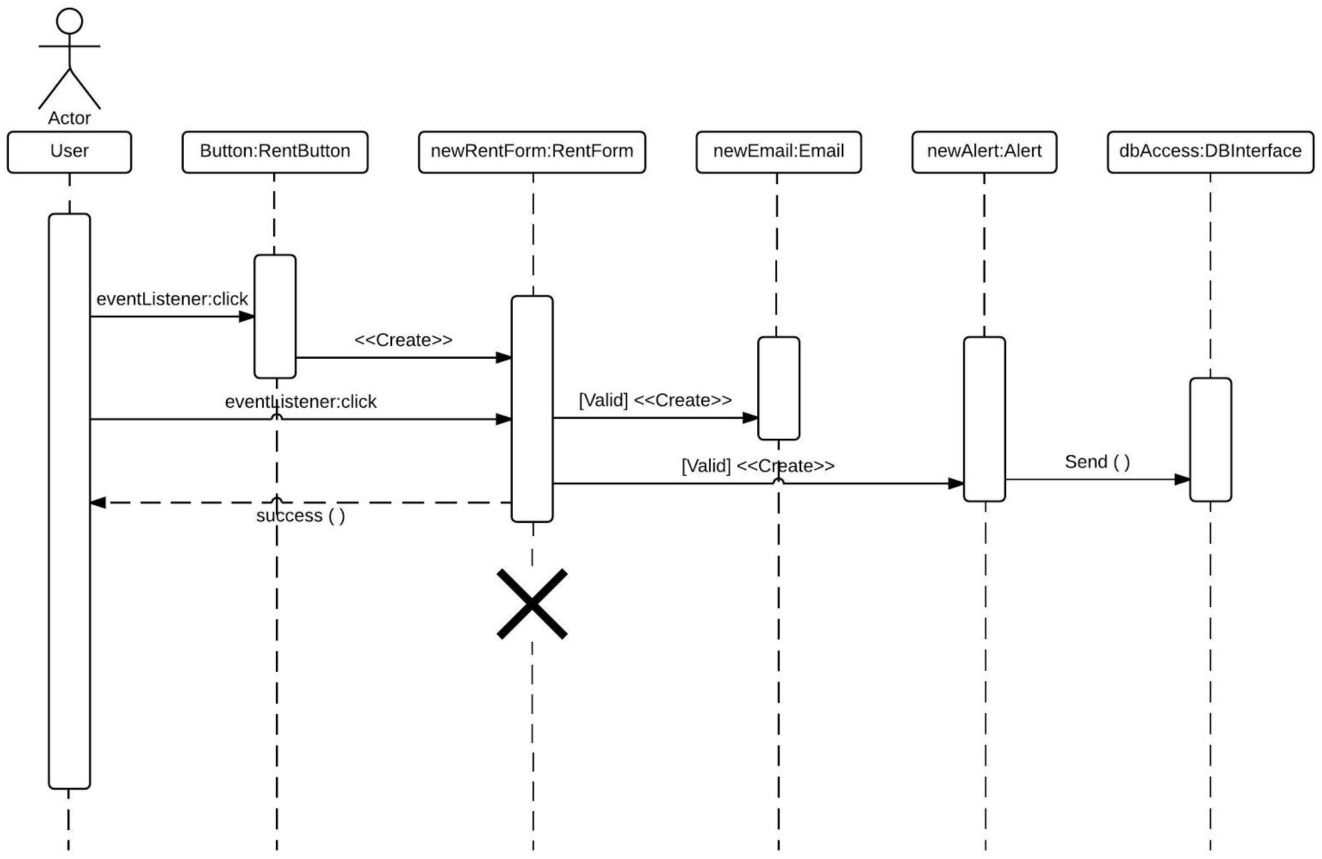
User interaction #1: Perhaps the most important user interaction that will take place on Neighbor is the ability for a user to post a new item to the website. A user will click on a button saying 'Post An Item', which will trigger a form (UI widget object) to appear. The user will then fill out the appropriate fields of this form and click on another button saying 'Submit To Neighbor'. In doing so, a new Item Object will be created and the information on the form will be posted to our backend database in JSON form (thus storing our object item in our database for later retrieval) as well as whatever pictures may exist. Note* It is likely we will have a class that interfaces with our database. Below is a sequence diagram for this user interaction:



User interaction #2: The next user interaction that is pertinent is an ability to search for items on Neighbor. The user can enter in a word into the search bar at the top of their home screen and click search (this word is treated as a regex and compared to the titles of all items currently on Neighbor) so an interaction with the database occurs. This will bring us to another web page which will generate a list of Item Objects that matched the user's search. Below is a sequence diagram for this user interaction:



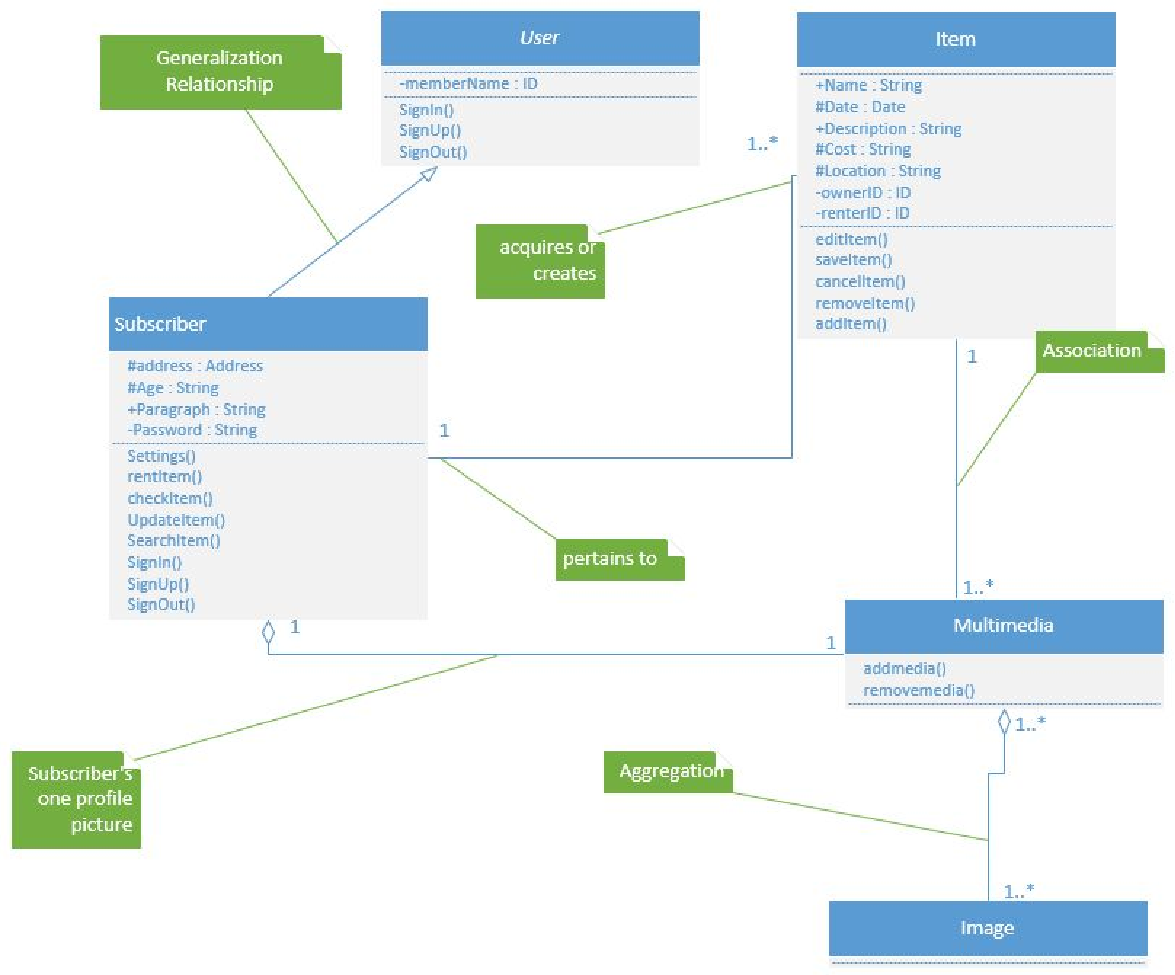
User interaction #3: Another important interaction is the ability to make an offer on an item that is in the search results. The user should be able to click a button saying ‘Rent!’ which will send a message to our system stating to create a UI widget object (another form) which once again the user will fill out with the appropriate information. When the rentee clicks ‘Submit Offer’, an interaction will take place with our database and store this Offer object inside our database. This acts as a message sent to the renters’ home screen (although modeling the owners interaction with this new offer would be a different use case). When this Submit Offer button is pressed, it stores the above information in our database as we just described, however it also generates an e-mail that will be sent to the owner of the item. Below is a sequence diagram for this user interaction:

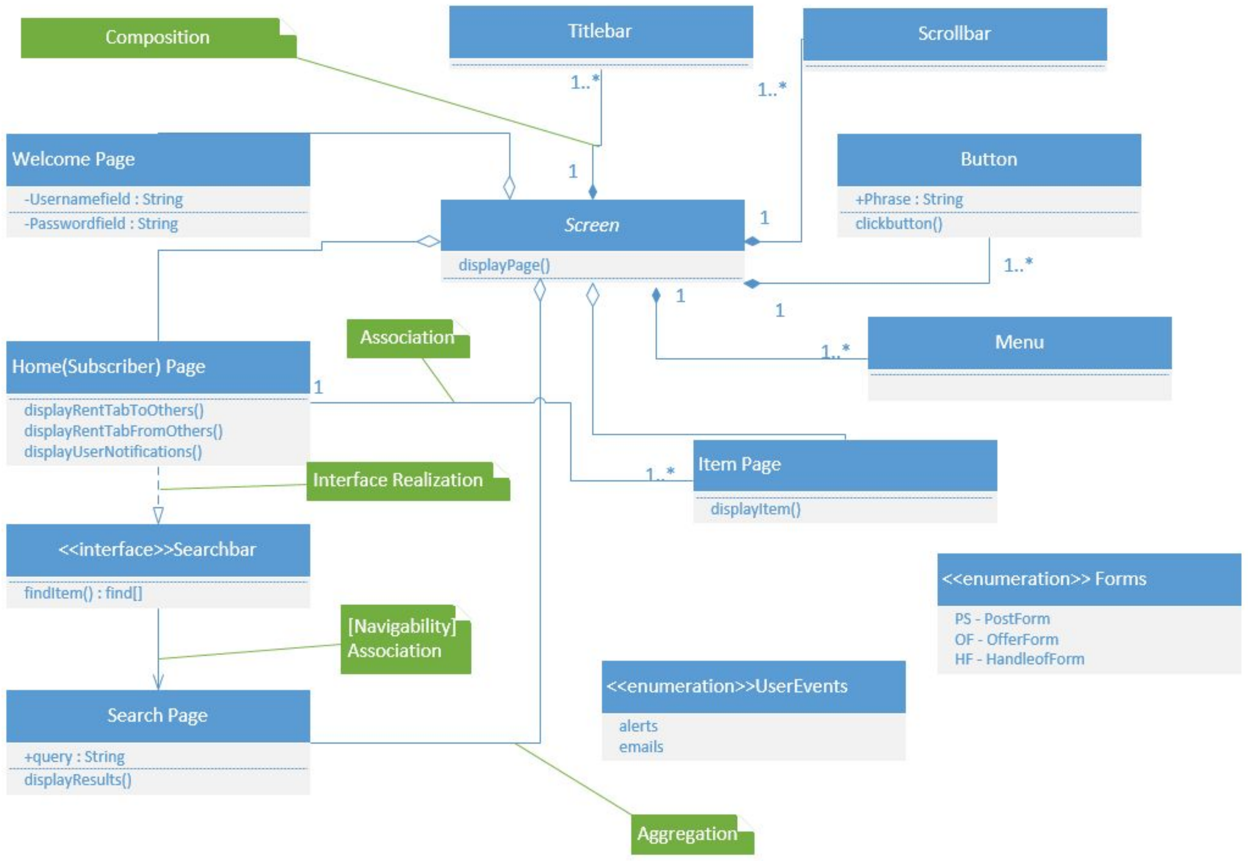


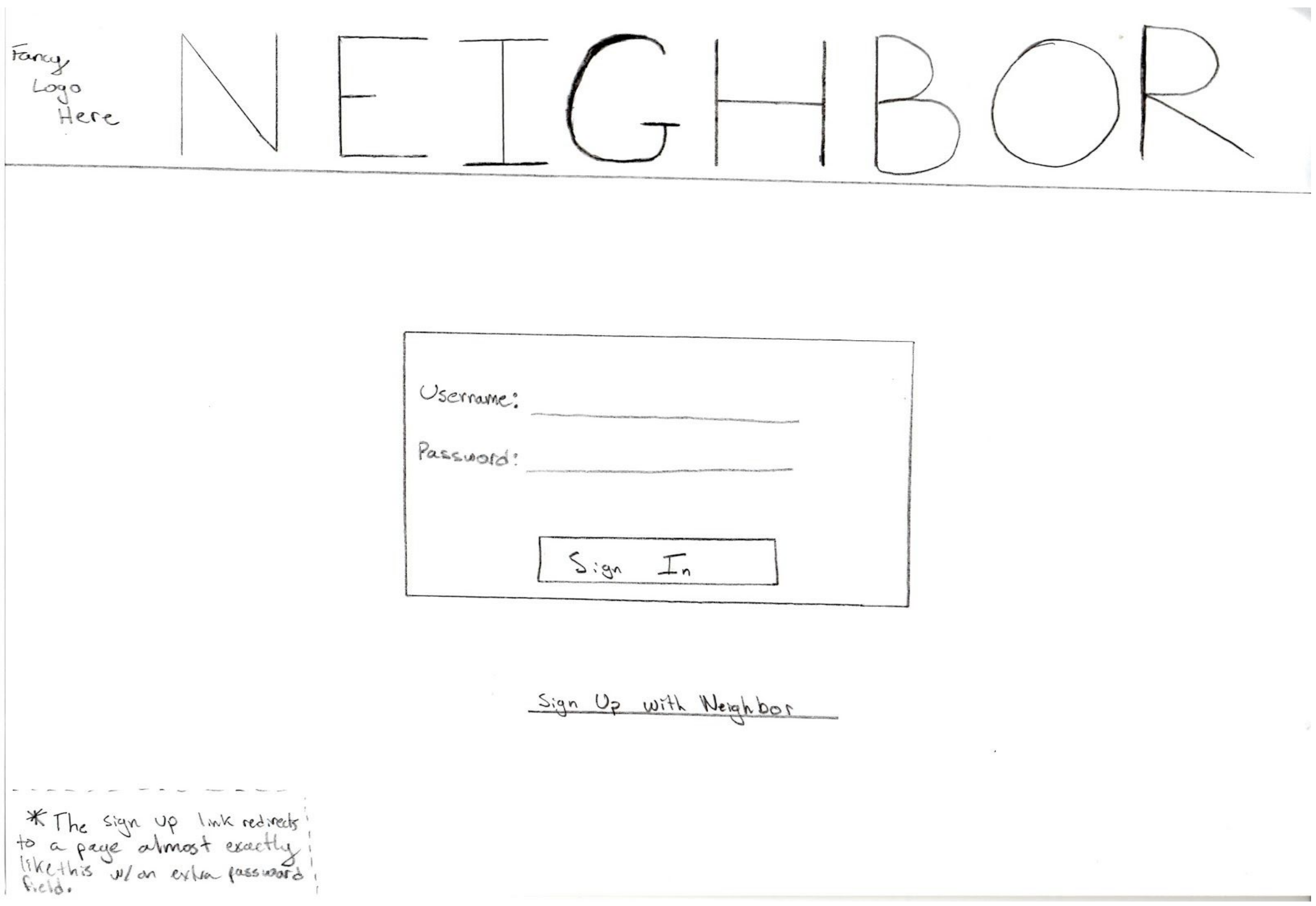
Class Diagram:

Must contain:

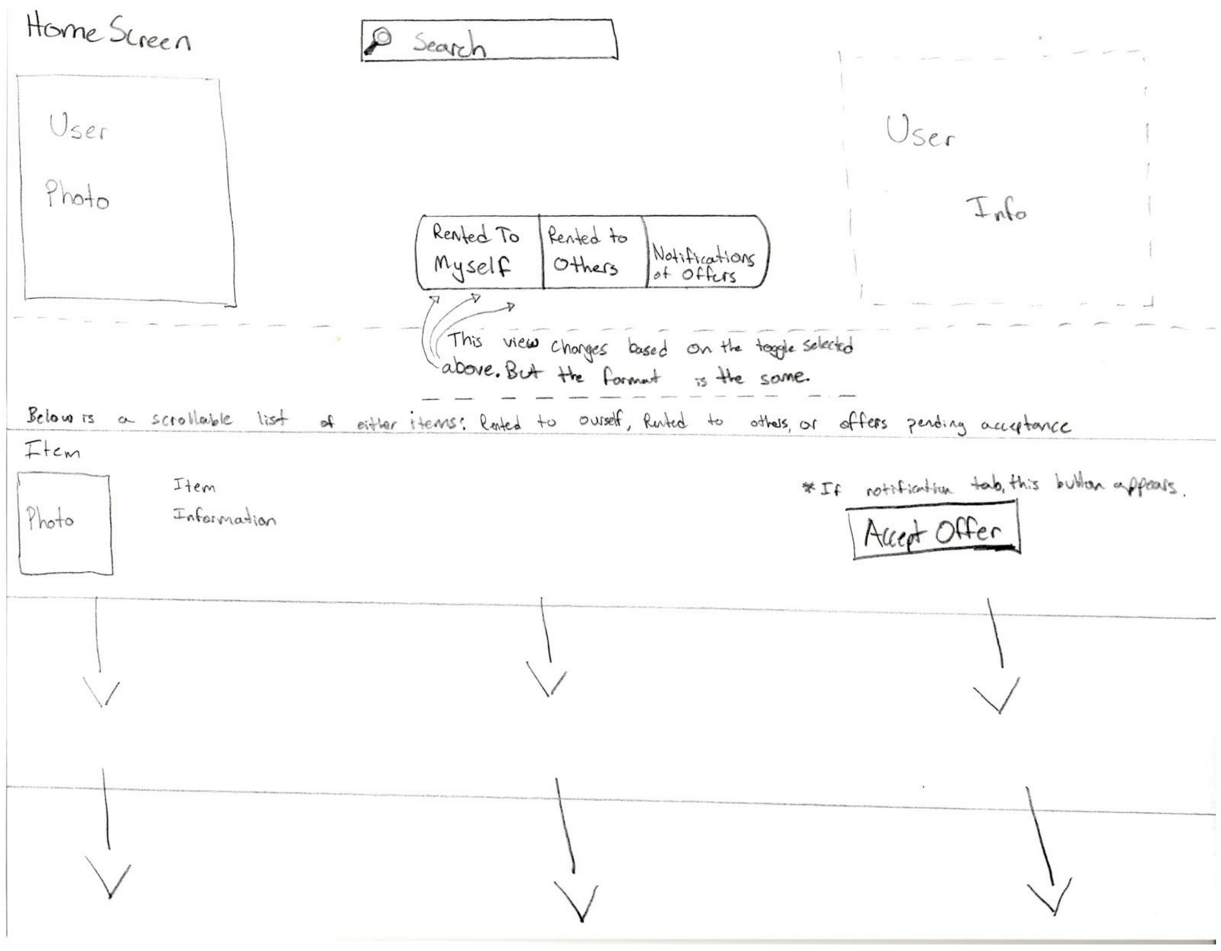
- What relationships they have ✔
- What are their attributes and (public) methods ✔
- What design patterns you may already know about are present in your design ✔
- Show visibility modifiers and relationships between the classes (use either notation presented). ✔




Composition
Titlebar
Scrollbar
1..*
1..*
Welcome Page
-Usernamefield : String
-Passwordfield : String
1
Button
+Phrase : String
clickbutton()
Screen
displayPage()
1
1..*
1
1
Menu
1..*
Association
Home(Subscriber) Page
1
displayRentTabToOthers()
displayRentTabFromOthers()
displayUserNotifications()
Interface Realization
Item Page
1..*
displayItem()
<<interface>>Searchbar
findItem() : find[]
<<enumeration>> Forms
PS - PostForm
OF - OfferForm
HF - HandleofForm
[Navigability]
Association
<<enumeration>>UserEvents
alerts
emails
Search Page
+query : String
displayResults()
Aggregation



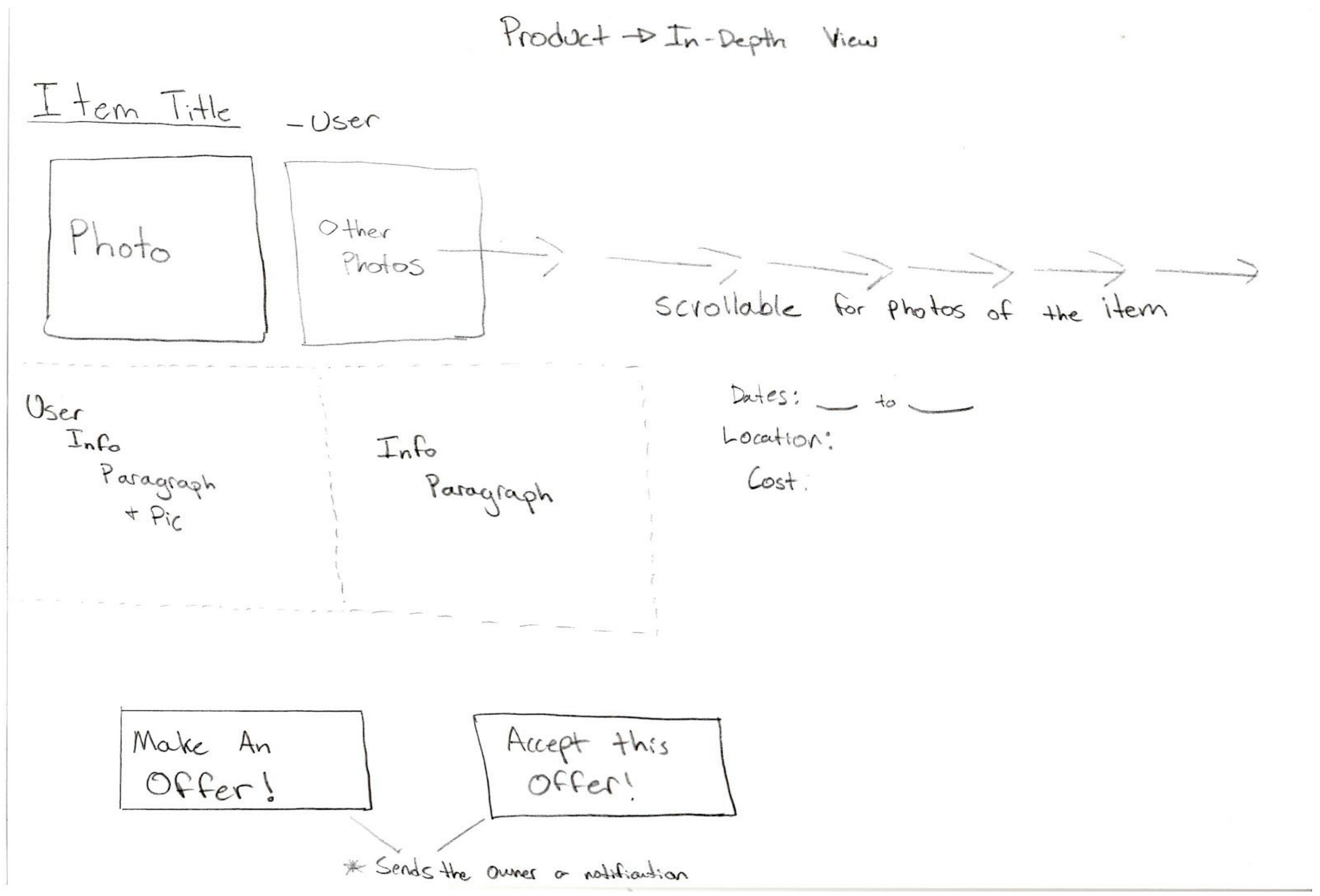
*The sign up link redirects to a page almost exactly like this w/ an extra password field.

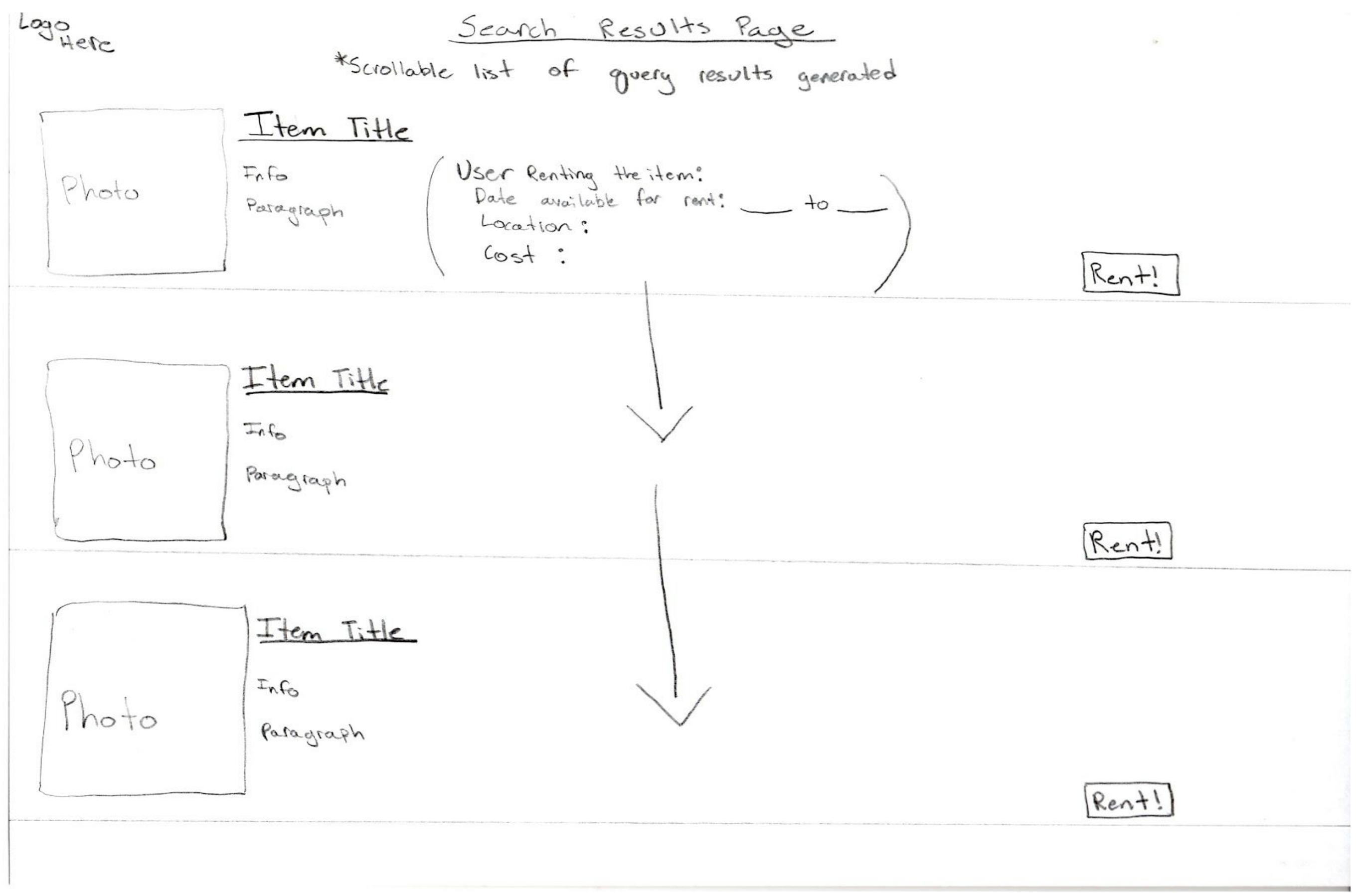


This view changes based on the toggle selected above. But the format is the same.

Below is a scrollable list of either items: Rented to ourself, Rented to others, or offers pending acceptance

*If notification tab, this button appears.


Product → In-Depth View
Item Title
– User
Photo
Other Photos
scrollable for photos of the item
User Info Paragraph + Pic
Info Paragraph
Dates: ___ to ___
Location:
Cost:
Make An Offer!
Accept this Offer!
* Sends the owner a notification


Logo Here
Search Results Page
*Scrollable list of query results generated
Item Title
Photo
Info
Paragraph
User Renting the item:
Date available for rent: ___ to ___
Location:
Cost:
Rent!
Item Title
Photo
Info
Paragraph
Rent!
Item Title
Photo
Info
Paragraph
Rent!