CSC 0648 Software Engineering Fall 2020

Online Buy/Sell/Exchange Website for SFSU students

MileStone 2

Team 05

Team members:

Naweeda Qeyam nqeyam@mail.sfsu.edu

Joshua Hansen

Dang Anh Tu Nguyen

Manuel Hernandez

Marvin Thai

Austin Bernard

Date Submitted:	**15/10/2020**
Date Revised (after instructor comments):	

Table of Contents

Executive Summary:

Our Service

Students everywhere have always required specific resources to meet their educational goals. These resources vary widely but are quite specific to the role of being a student at a university, and much of these needs go unmet throughout the career of the student. To name just a few examples, faculty, staff, and students are often looking for things such as tutoring, cheap textbooks, class-specific tools like iClickers, or perhaps even furniture for those who are moving to live close to or on their university of choice. All these needs involve the exchange of goods and services from one student to another, meaning that there is a large untapped market for a specialized marketplace where students can exchange goods and services amongst each other quickly and safely. Our service aims to meet this need and bridge the gap in this market.

The website we are developing will serve as a student marketplace for SFSU students and staff to exchange items or services safely, quickly, and easily between each other. The website will provide a platform for students to easily sell and buy things like textbooks, tutoring services, electronics, miscellaneous accessories, furniture, and a variety of other things. Although with services like Amazon or eBay today, it is already quite easy to find items online, our website brings a novel approach of ease-of-use, quickness, and variety to the online trading world. Students will be trading in a marketplace that is specific to their university, allowing for a much faster transaction to take place. Students can also exchange items for other items, giving them more freedom and flexibility with how they value and exchange the items they choose to sell. Students will find it easier to search for what they need, as materials can be directly linked to and categorized by the courses that require them, if any.

This website will also help instructors, which is giving them the opportunity to provide resources to their students directly. Professors can easily assign books under their classes and names, so students can search for their class materials by course or instructor name. This will also assist students looking to sell used textbooks, as the marketplace will ensure their used books are easily found by students looking for them.

Given that we are a team of students developing this site, we are most qualified to understand the needs of students overall. We have firsthand experience with what students need, and what others can offer, and we are intimately familiar with what makes an online marketplace easy to navigate and use, being customers ourselves. With our personal experience and expertise, we plan to deliver a resource that stands out above any other online marketplace today, in the spirit of providing an unmatched resource catered towards students, teachers, and the educational community as a whole.

List of main data items and entities

1. **Communication:** A conversation between buyer and seller about posted items
2. **Products:** Physical material or non-tangible items that will be sent/received between the 2 buyers and sellers. (eBooks, Calculator, laptop, software, etc.)
3. **Services:** An agreement between two users where one user (buyer) pays the other user (seller) to execute tasks for them. (i.e. tutoring, dog walking, note taking)
4. **Posting:** Provide product page, or service to be agreed upon through Communication between Buyer and Seller
    - Product title: The title of the product for posting
    - Product description: The description of the product and the owner
    - Product price: price of the product is set by the sellers
    - Product category
        - Course ID required if category belongs to Books category
    - Product image
5. **Unregistered Users:**
    - Can browse through the listings of services and products on the website but cannot purchase nor create posts.
    - Can create an account
6. **Registered Users:** Registered users can post products/services to sell and pay for them in vice versa.
    - Registration Record (METADATA)
        - Registration Record: User SFSU id to validate students.
        - SFSU email address: User SFSU email to validate student. (@mail.sfsu.edu)
        - Name: How the user is addressed via communication.
        - Password: confidentiality for the users information encapsulated in their account.
7. **Admin:** Full system control of the website to oversee all traffic and usability.
    - Suspend or ban users from the website.
    - Has the capability to approve posts.
8. **Message:** Communication built between two users.
    - Product ID: The message tied to the product listed that facilitates what product they are talking about.
    - Title: The product/service title that reflects the Product ID.
    - Date: The date the message was sent.
    - Text: Plaintext of information given from the user with the inquiry.

Functional Requirements - prioritized

Priority 1:

Unregistered Users:

1. Unregistered users shall be able to browse the products/services on the website.
2. Unregistered users shall be able to search using a search bar.

Registered Users:

3. Registered users inherit all features of unregistered users.
4. Registered users shall be able to post items with all the data such as products/services with their username tied to the product, while being able to set the price of their product or service.
5. Registered users shall be allowed to add relevant course information about the school item they post for sale.
6. Registered users shall be allowed to remove their sales/services posts.
7. Registered users shall be able to message other users directly through the product post based on the item of interest.
8. Registered users shall be able to message back the exact buyers who initiated the message.

Admin:

9. Admin required to approve all posts that are requesting to be listed.
10. Admin shall be able to suspend and ban users.

System:

11. The system shall allow only SFSU students/employees/staff to create an account to buy and sell school supply as a user

Priority 2:

Unregistered Users:

12. Unregistered users shall be able to search for desired items listed using a filter.

Registered Users:

13. Registered users shall be able to adjust price after posting.
14. Registered users shall be allowed to post under their profile a list of services that they are providing.

System:

15. The system shall allow the users to personalize their profile, such as name, address.

Priority 3:

Registered Users:

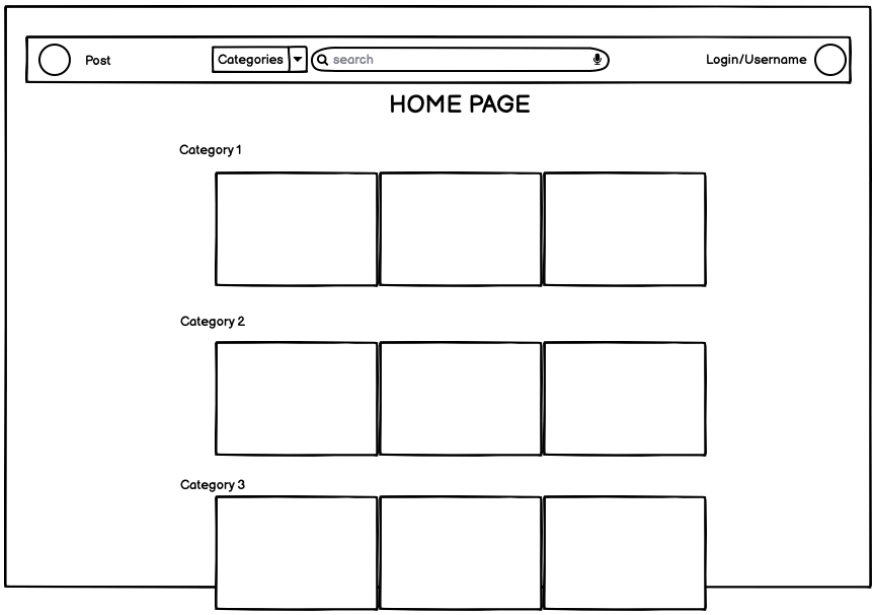
16. Registered users shall be able to coordinate delivery options through the portal.
17. Registered users shall be capable of uploading a profile picture.
18. Registered users shall be able to rate their seller or buyer experience.
19. Registered users shall be able to request for notification when a book is available from any seller.

UI Mockups and Storyboards

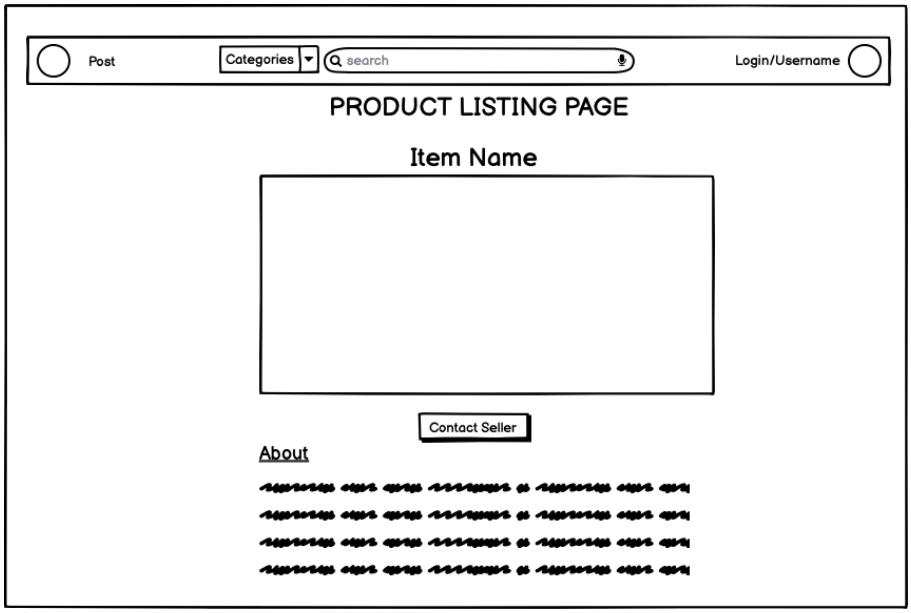
Use Case 1:

(1) Sarah is looking for tutoring by class on our website but finds some furniture that she could use for her new dorm when browsing the home page. **(2)** She is already logged in and navigates to that page and messages the seller by clicking the contact seller button. **(3)** This navigates to the messenger page and she messages the seller. She then wants to go back and search for the tutoring by class and clicks the category and selects tutoring and then inputs the course ID. **(4)** This takes her to the search results and she finds a post for her class and then clicks the post to bring her to the posting page. **(5)** She then clicks the contact seller button, and the site takes her to the messaging page and she **(6)** sends her message to the seller.

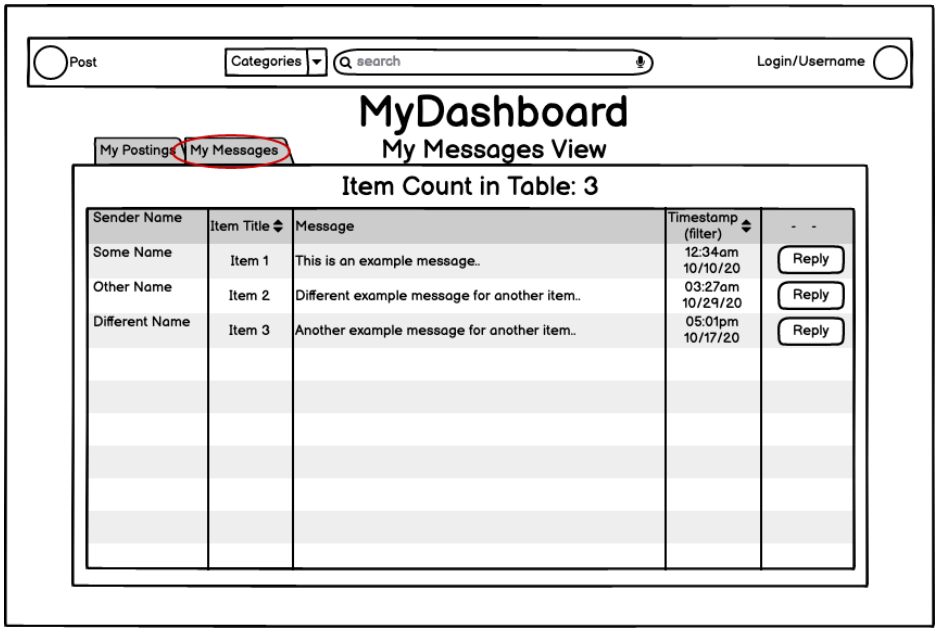
(1)



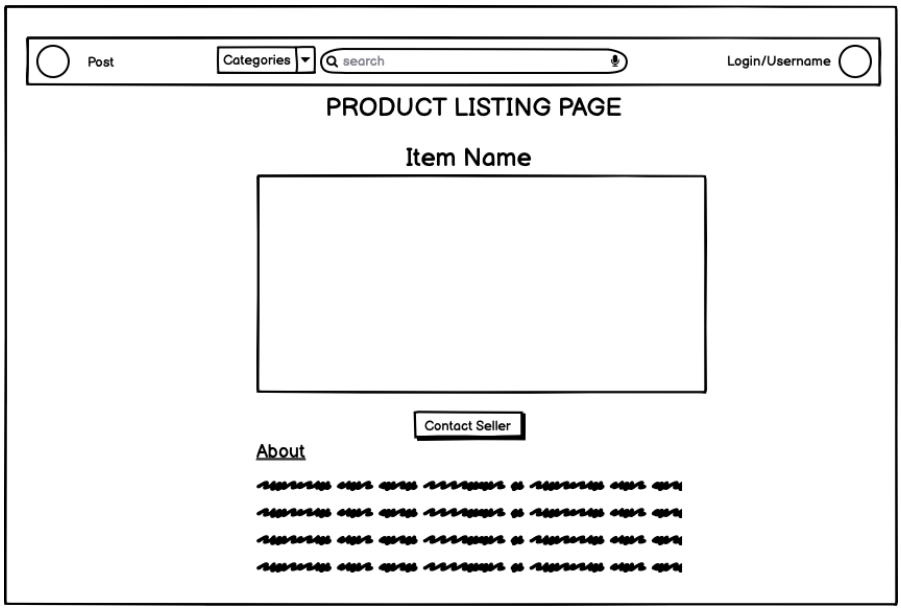
(2)


Post
Categories
search
Login/Username
PRODUCT LISTING PAGE
Item Name
Contact Seller
About

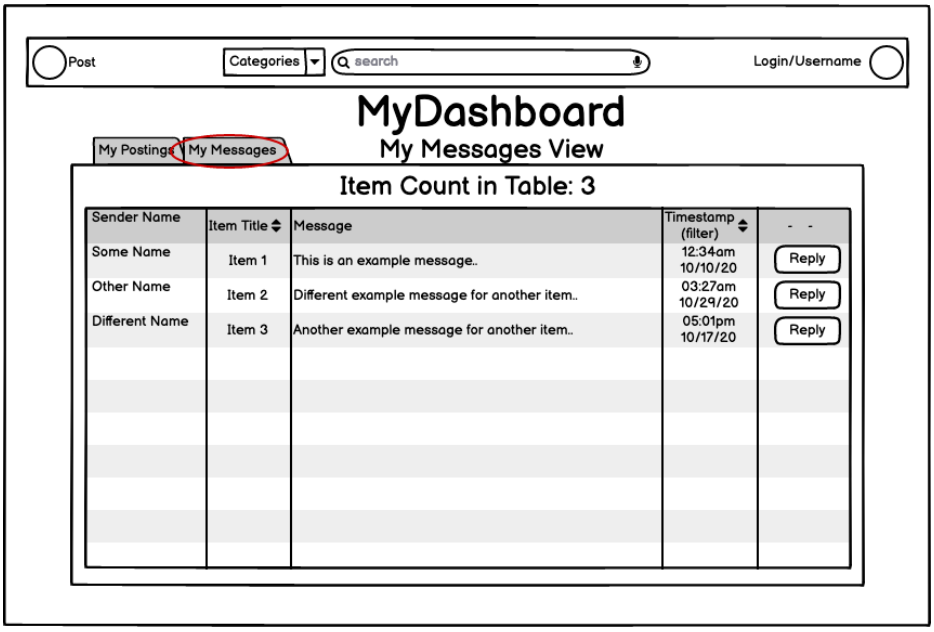
(3)


Post
Categories
search
Login/Username
MyDashboard
My Postings
My Messages
My Messages View
Item Count in Table: 3
Sender Name
Item Title
Message
Timestamp (filter)
Some Name
Item 1
This is an example message..
12:34am 10/10/20
Reply
Other Name
Item 2
Different example message for another item..
03:27am 10/29/20
Reply
Different Name
Item 3
Another example message for another item..
05:01pm 10/17/20
Reply

(4) (5)

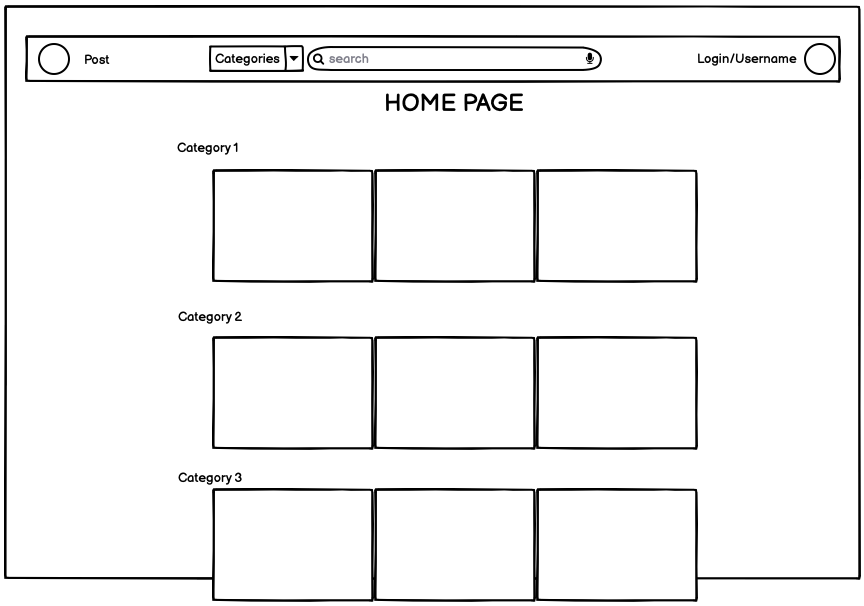

Post
Categories
search
Login/Username
PRODUCT LISTING PAGE
Item Name
Contact Seller
About

(6)

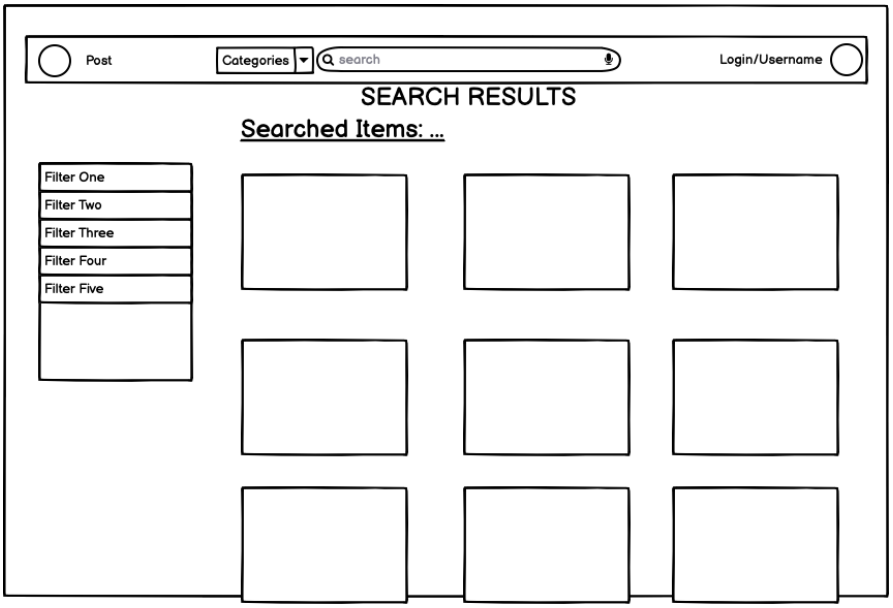

Post
Categories
search
Login/Username
MyDashboard
My Messages View
My Postings
My Messages
Item Count in Table: 3
Sender Name
Item Title
Message
Timestamp (filter)
Some Name
Item 1
This is an example message..
12:34am 10/10/20
Reply
Other Name
Item 2
Different example message for another item..
03:27am 10/29/20
Reply
Different Name
Item 3
Another example message for another item..
05:01pm 10/17/20
Reply

Use Case 2: (1) Tom wants to search for a book for his class. He goes to the home page. **(2)** He clicks the categories dropdown next to the search bar and types the course ID into the search bar. This takes him to the search results page, and he finds a post selling the book. **(3)** He clicks on this post and navigates to the post page. (**4)** He then clicks on contact seller but because he is not logged in, he is redirected to the login page. **(5)** After logging in he is taken to the messaging page so he can message the seller.

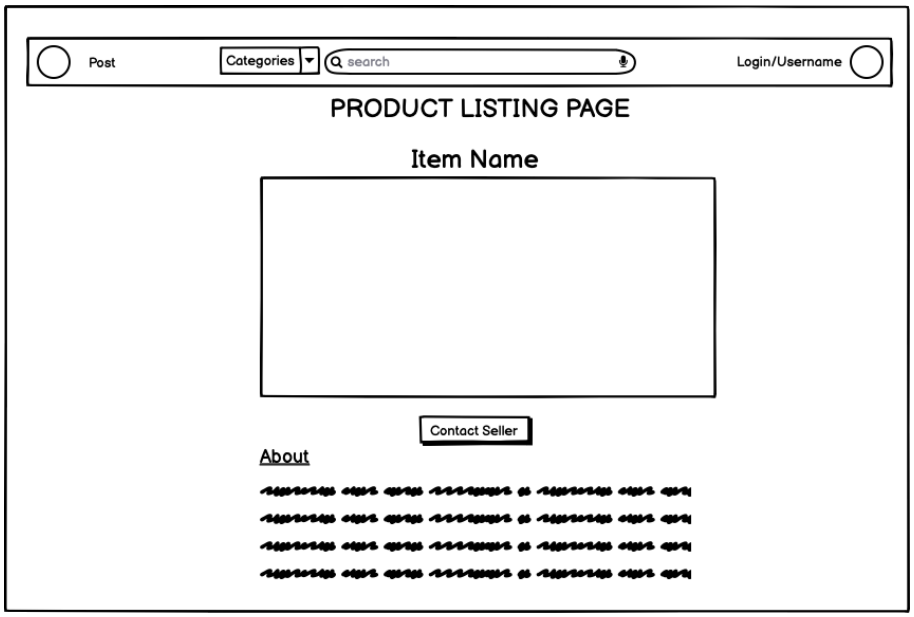
(1)



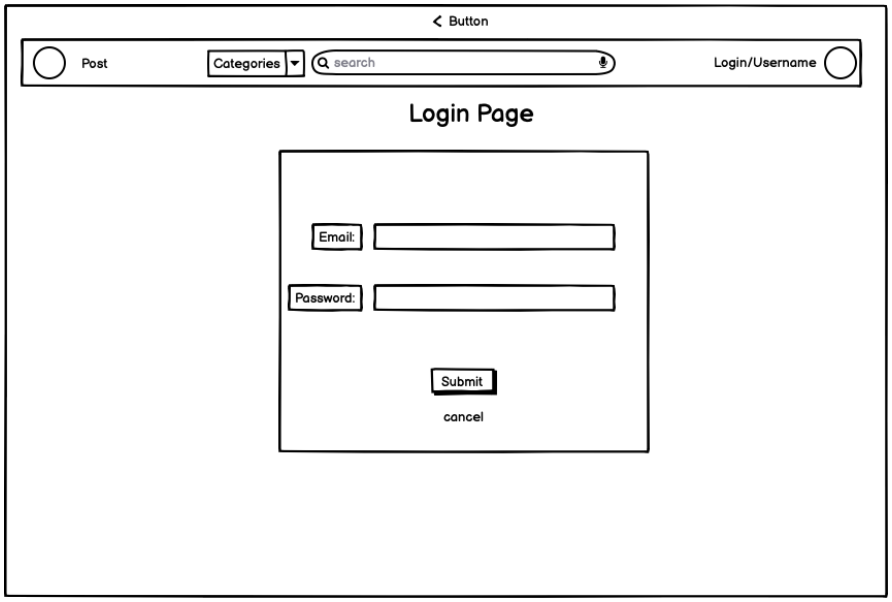
(2)



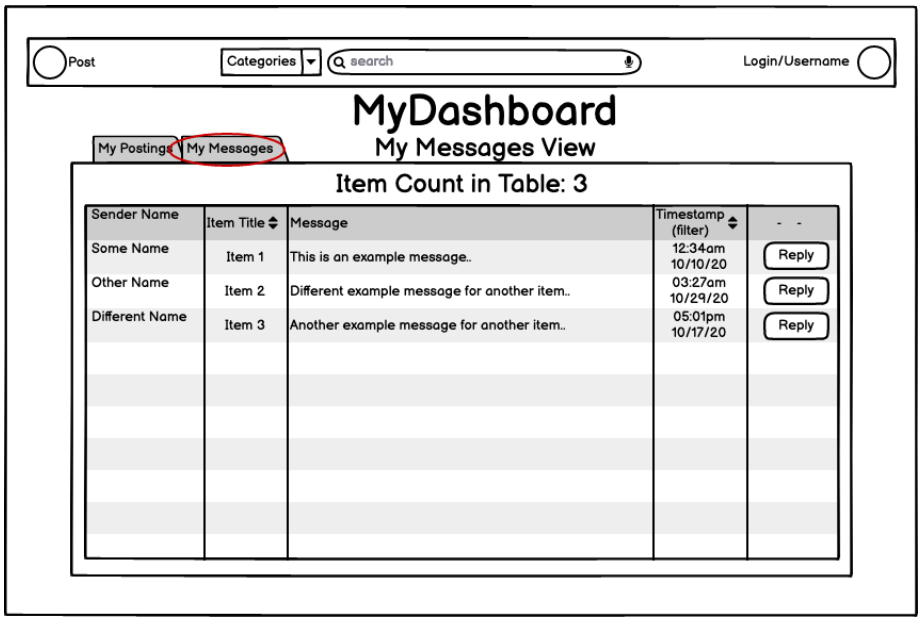
(3)


Post
Categories
search
Login/Username
PRODUCT LISTING PAGE
Item Name
Contact Seller
About

(4)

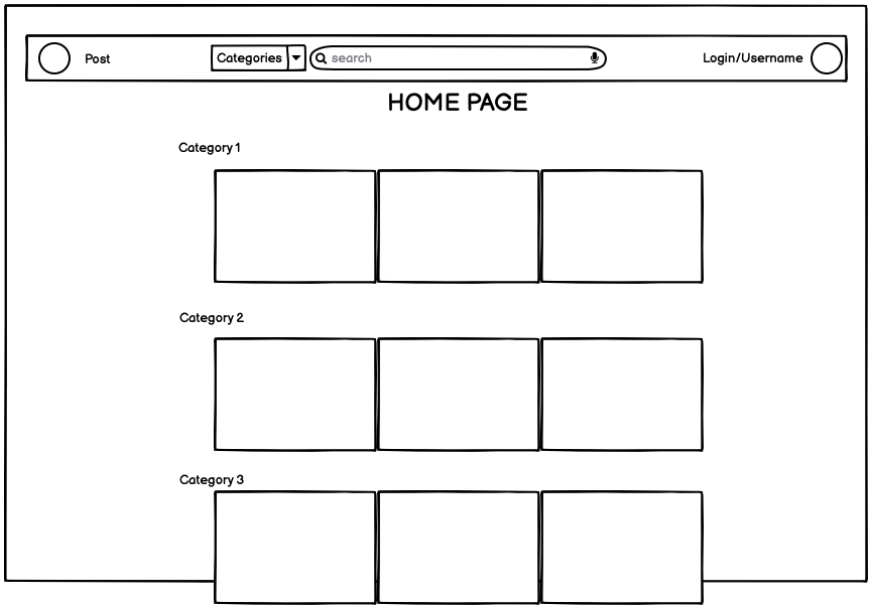

Button
Post
Categories
search
Login/Username
Login Page
Email:
Password:
Submit
cancel

(5)

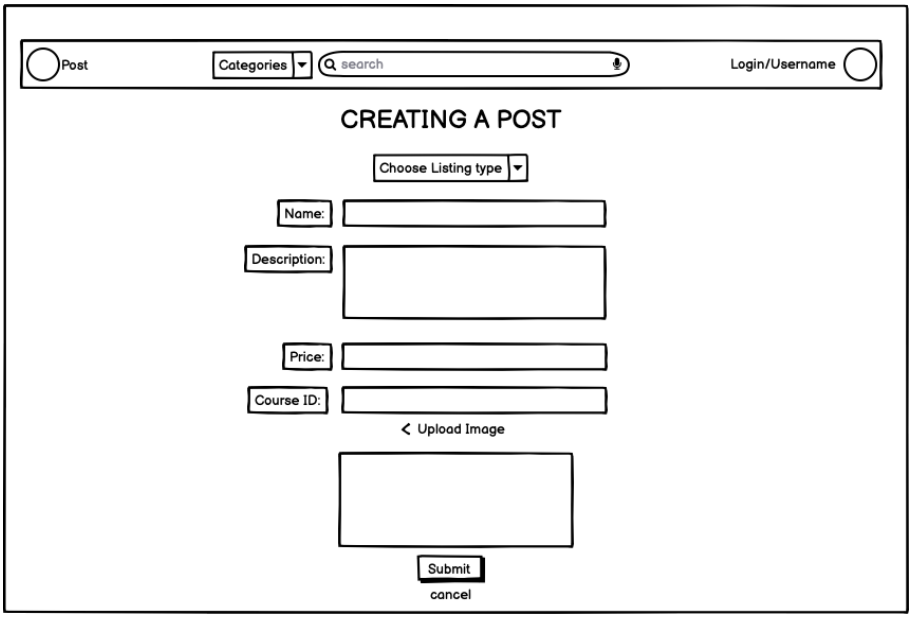

Post
Categories
search
Login/Username
MyDashboard
My Postings
My Messages
My Messages View
Item Count in Table: 3
Sender Name
Item Title
Message
Timestamp (filter)
- -
Some Name
Item 1
This is an example message..
12:34am 10/10/20
Reply
Other Name
Item 2
Different example message for another item..
03:27am 10/29/20
Reply
Different Name
Item 3
Another example message for another item..
05:01pm 10/17/20
Reply

Use Case 3: Joseph is logged into the site and wants to sell his extra books. **(1)** He clicks the ‘Post’ button on the homepage in the top left corner and this takes him to the **(2)** page to create a post. When he chooses the listing type of textbooks, a course ID field pops up in the bottom of the form and is now a required field. He fills out his information and then clicks the create button and his post is created. **(3)** He then is directed to a confirmation page and then goes to his ‘My Dashboard’ page which shows his postings.

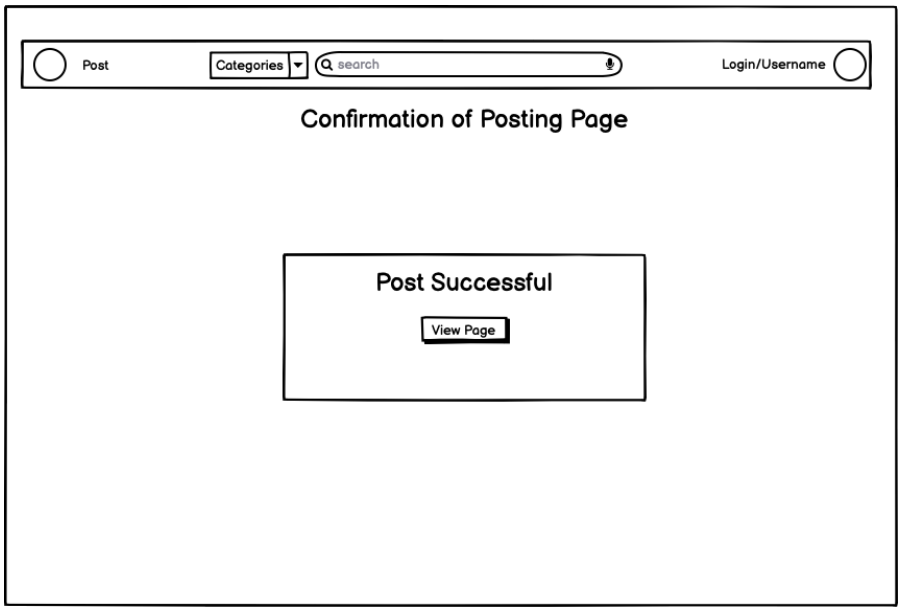
(1)



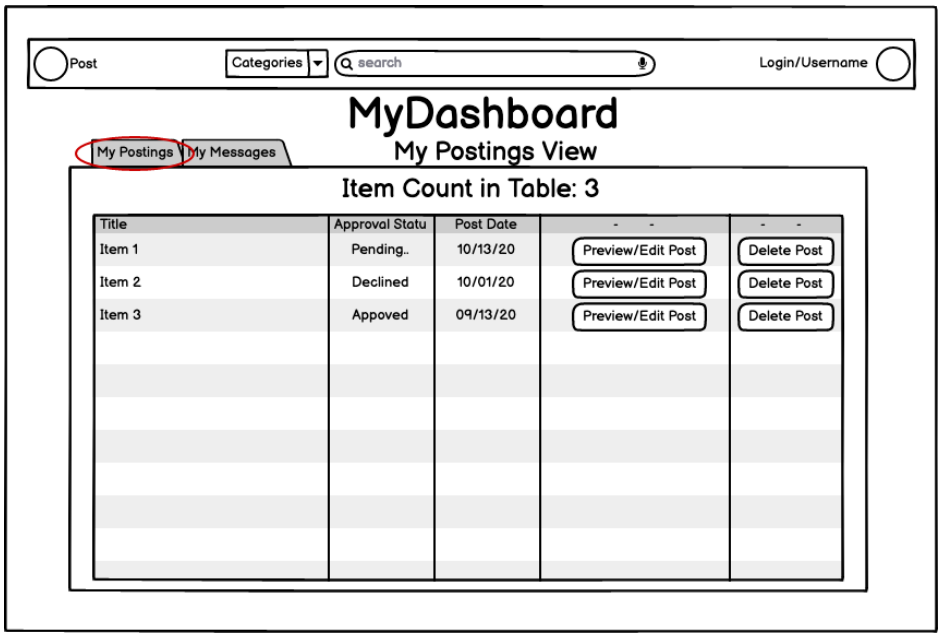
(2)



(3)

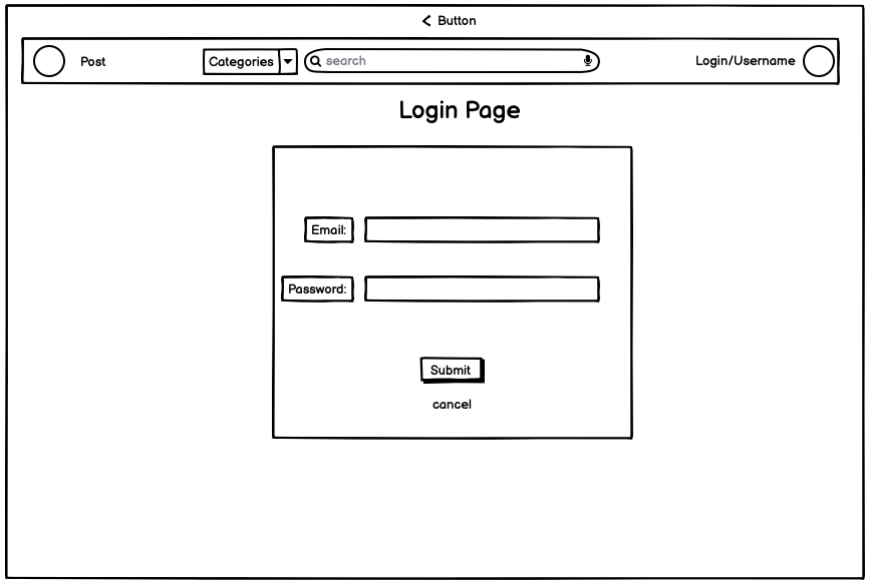

Post
Categories
search
Login/Username
Confirmation of Posting Page
Post Successful
View Page

(4)

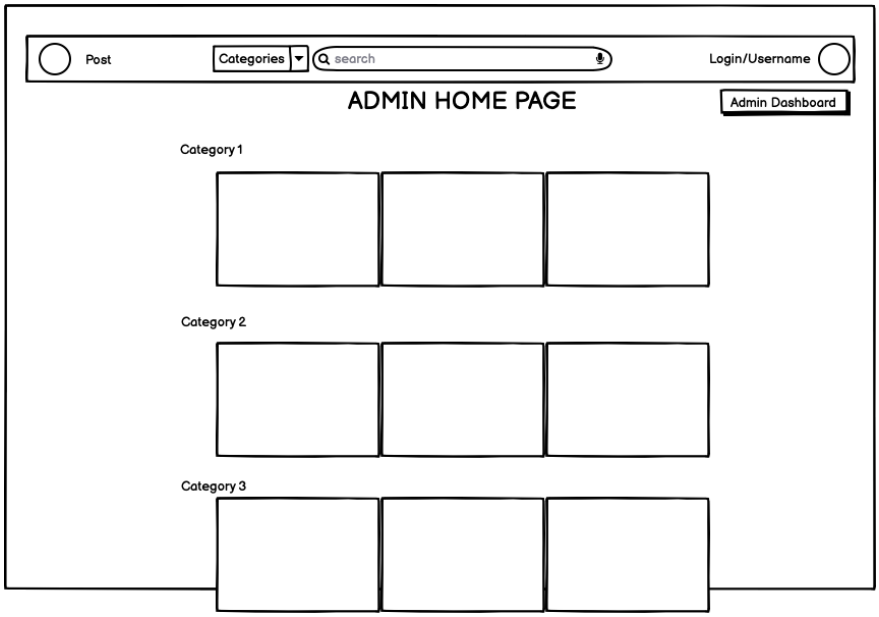

Post
Categories
search
Login/Username
MyDashboard
My Postings
My Messages
My Postings View
Item Count in Table: 3
Title
Approval Statu
Post Date
Item 1
Pending..
10/13/20
Preview/Edit Post
Delete Post
Item 2
Declined
10/01/20
Preview/Edit Post
Delete Post
Item 3
Appoved
09/13/20
Preview/Edit Post
Delete Post

Use Case 4: Kim needs to approve some pending posts as an admin so **(1)** she logs into the site and is taken to the **(2)** home page with an extra option in the nav bar to navigate to the admin dashboard page. **(3)** When on the admin dashboard page, **(4)** she clicks the image of the pending posts and a preview for the post shows on the right side of the page. She then can close out and choose to approve or deny the post.

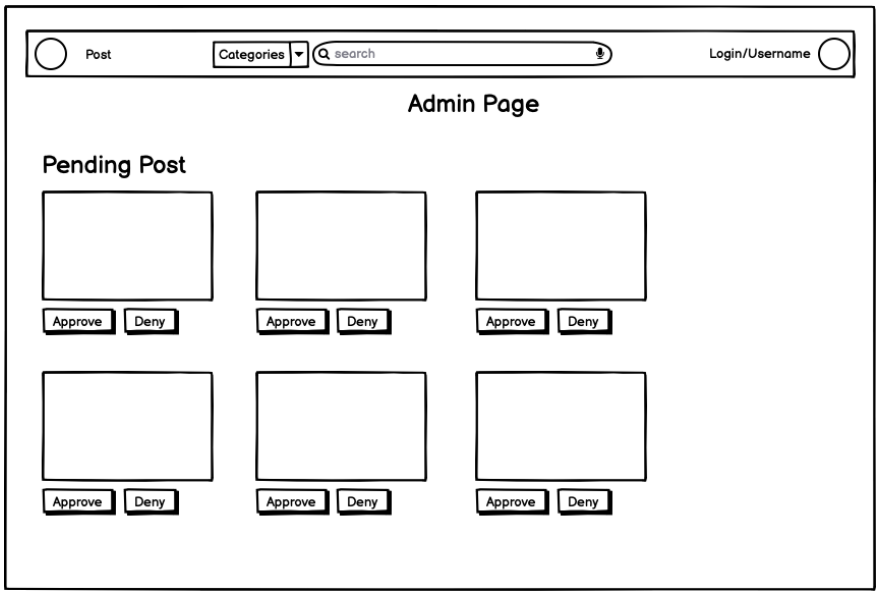
(1)



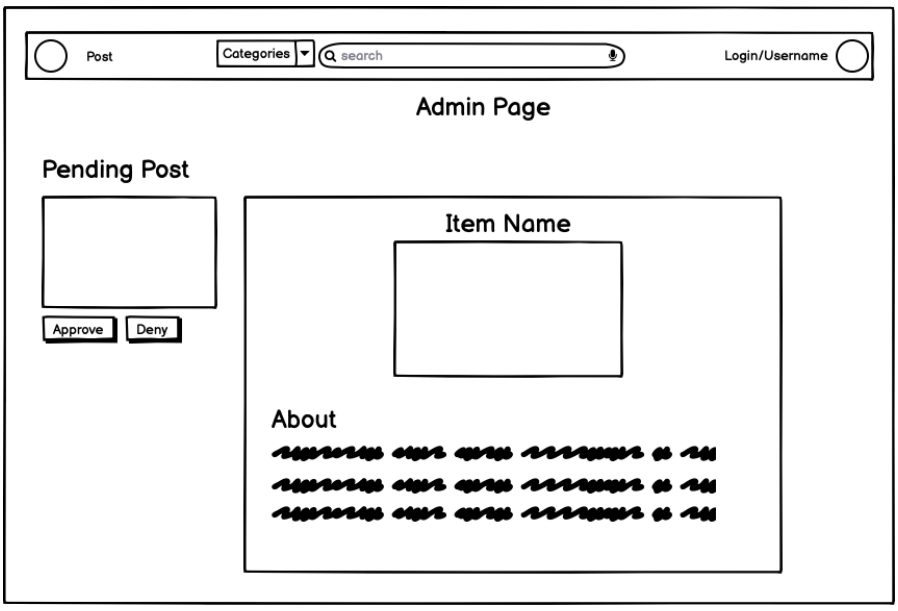
(2)



(3)


Post
Categories
search
Login/Username
Admin Page
Pending Post
Approve
Deny
Approve
Deny
Approve
Deny
Approve
Deny
Approve
Deny
Approve
Deny

(4)


Post
Categories
search
Login/Username
Admin Page
Pending Post
Approve
Deny
Item Name
About

High level Architecture, Database Organization

User table:

- email\ (Primary Key)
- password
- firstName
- lastName
- image

Admin table:

- Email (primary key and foreignkey)

Posting table:

- postID (primary Key)
- email (unique)
- Title
- Description
- Price
- Category
- image

Messages table:

- Id (primary key)
- Mid (foreign key)
- messageBody
- Sender
- timestamp

Messages handler table:

- Mid (primary key)
- postID (foreign key):
- Author
- inquiry

Category table:

- postingID (foreign key)
- Category_name
- CategoryID (primary key)

Media storage

The images in our application will be stored in the flask application in a static folder. In the database, we will save the relative path of each posting. The acceptable formats are: .png, .jpg, .gif.

Search/Filter Architecture and Implementation

When the user enters in keywords such as “books for physics” - the search bar will grab each part and create it as such “searchQuery = "SELECT * FROM Categories C, Posts p WHERE C.cid = p.Categories_cid and (p.title REGEXP '{}' or p.Description REGEXP '{}');".format(info, info)

High level API and Main Algorithm:

validUser ():

- this function will take the username and password of a user attempting to log in and will try to validate if the information matched the information in the database. If the information is validated the process will let the user into the main page, else it will tell the user the information does not match.

findUser ():

- This works the same as validUser

findUserType ():

- This function only takes a user's username, and it sends back the type of user it is. Ex is inputted "ortiz2019" into the function, the function will say that "ortiz2019" is an "admin".

addUser ():

- It stores the user sign up information into the database.

getUserAccount ():

- This will send back the current user account(username) signed in.

setUserAccount ():

- The input of this function is a username, and it will be assigned to the current user.

getUserPassword ():

- This function will send back the current user's password.

setUserPassword ():

- This process takes in a password type and sets it as the password of the current user's account.

getUserName ():

- This will return the current user's username.

getUserEmail ():

- This will return the user ID assigned during registration. This will check if the ID is in the database.

addPosting ():

- This function takes a number ID of a posting and adds it to a list of postings ready to be posted to the mainboard and the database.

removePosting ():

- This function will remove a posting from the list that is ready to be posted on the mainboard and the database.

editProduct ():

- This function takes in a posting posted as an input, once the posting input is found, it lets us call other functions which allows the user to edit and update information about the posting posted

getPostingID ():

- This function takes an item already posted as input and sends back the ID of that item.

getPostingTitle ():

- This function takes in a posting ID and if the ID is found then it sends back the title of the posting

setPostingName ():

- This function takes in a posting ID and if the ID is found then it sets the name of that posting.

getPostingType ():

- This function takes in a posting ID and if the ID is found then it sends back the posting category

findBookByCourse ():

- This process takes in a course name as input and if the course is found in the database it will send back information about the book that use by that course.

findServices ():

- This process takes any type of service as input and if the service is found in the database it will send back information of all the services available of such type.

API endpoints:

Action: Sign-in button

POST login/:

{email: string, password}

○ **Action: Signup button**

POST signup/

{email: string, password: string, firstName: string, lastName: string}

○ **Action: Search button**

POST search/

{name: string, category: name}

○ **Action: Search page result**

POST search/

{name: string, category: name}

○ **Action: Create a product/ service post**

POST /posting {description: string, price:int, category:string }

○ **Action: Check a product page**

POST /posting/: postID

{email: string, postID:int}

○ **Action: User send a message through a post**

POST /posting/: postingID/message

{email: string, message: string}:

Actual key risks

Skill Risks:

- Team needs to learn API.
- Team members need to familiarize themselves with AWS tools.
- Team needs to learn how to work with Flask

Technical Risks:

- CPU/Server utilization using Amazon’s free tier of AWS
- Github usage when working on multiple parts of the project at once

Team Risks:

- Frontend & backend teams agreeing on a common way to connect UI with backend.
- Pacing the work so that neither the frontend nor backend team will not be waiting on the other to push new code.

Project management:

To successfully ship Milestone 2, we set up and provided our team with a template for meetings notes that our team lead would fill out prior to each meeting. The template consisted of our meeting agenda, tasks that we need to fulfill, various miscellaneous notes, and any issue we had regarding the implementation and completion of Milestone 2. We are also using a tool called Monday.com, which has made it easier for project management for us. It allows us to be able to view all our tasks and deadlines, and it is extremely simple to track the status of our tasks on the website dashboard. Frequent and efficient team meetings multiple times per week have also been critical to allowing us to maintain consistent progress toward completion of the Milestone.