Exchange4Students Milestone 5 Report

Document of Design Updates

SSW 322 Group 2

James Grant, Justin Phan, Jacob Gelman-Funk, Edmund Yuen

May 6th, 2025

Introduction/Summary Section

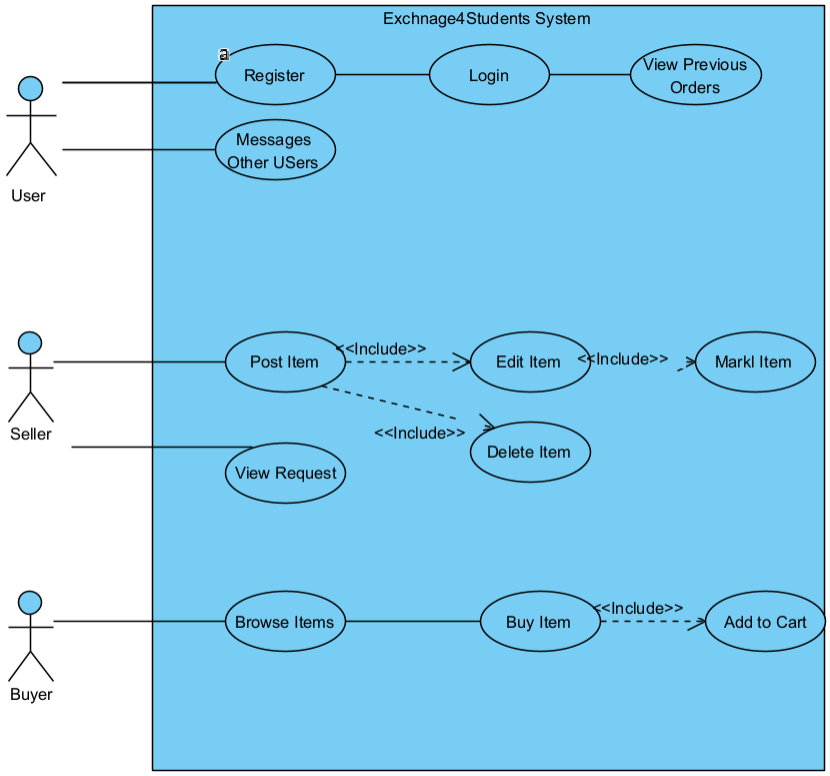
Since our previous milestone, this milestone focuses mostly on improvements to UI/UX and new features for buyers like canceling an order if the item is not sold yet and viewing their order history and status. More specifically, our cart and checkout UI were improved and features to improve user experience include showing passwords when registering/logging into our application for better accessibility or allowing users to clear request notifications. Also, more improved and specific error messages to what fields users are missing when registering, logging in, or checking out for better accessibility and help. For example, if they are missing a username and email they will receive an error message saying so or other combinations of other fields. Our seller page saw several UI updates as well, including improving our item posting form to only show the user specific fields per category and simplifying the landing page. Finally, for our buyer page, our item browsing UI has also been improved, as well as some refinements to our search functionality.

Team Collaboration

- James Grant: During this project, I contributed to:
    - Implemented cancel order function
    - Added base code for order history
    - Implemented function for buyers/sellers to ‘clear’ requests/notifications
    - Updated documentation
- Justin Phan: During this project, I:
    - Improved form validation for checkout, login, and register page

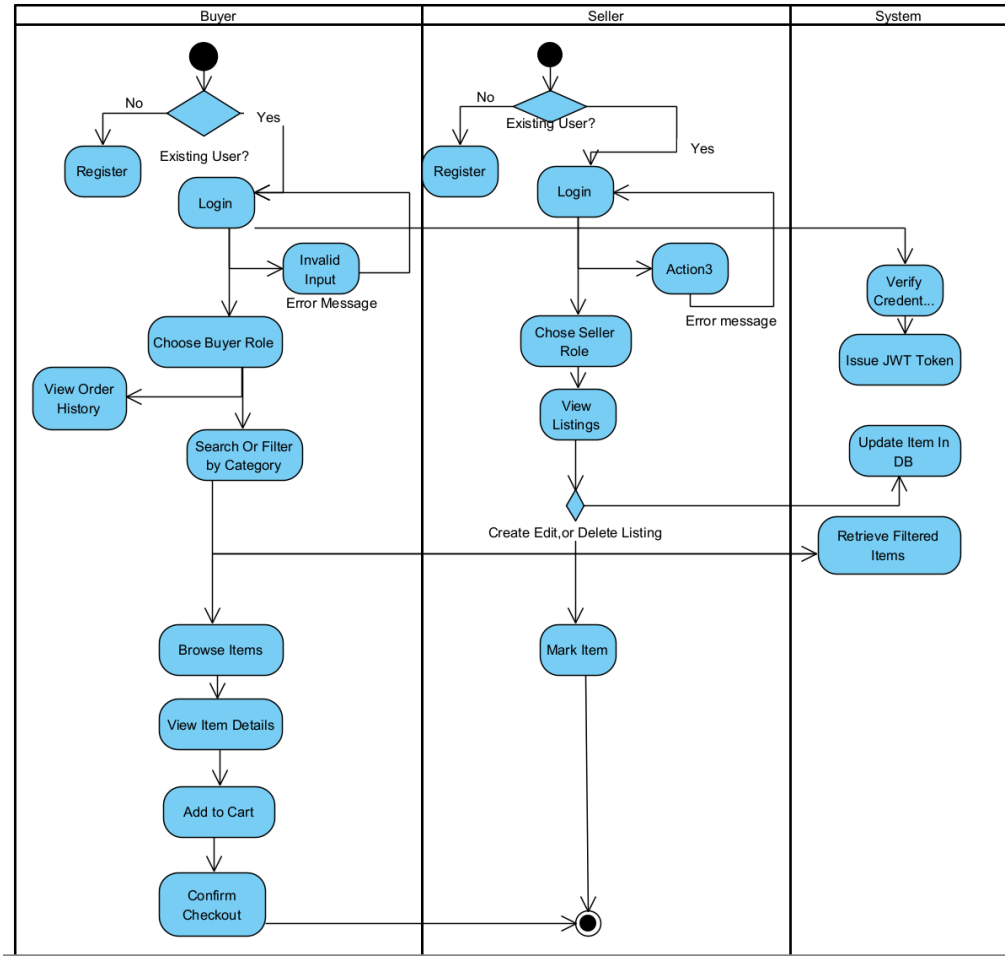
    - Implemented option to show password when registering a new account or logging in
    - Updated class diagram to reflect new entities
- Jacob Gelman-Funk: During this project I:
    - Improved existing UI Elements
    - Updated Use Case and Activity Diagrams
- Edmund Yuen: For this milestone I contributed:
    - Updated browsing UI
    - Item posting update based on item type
    - Order history
    - Updated checkout and cart UI

Updated Use Case Diagram



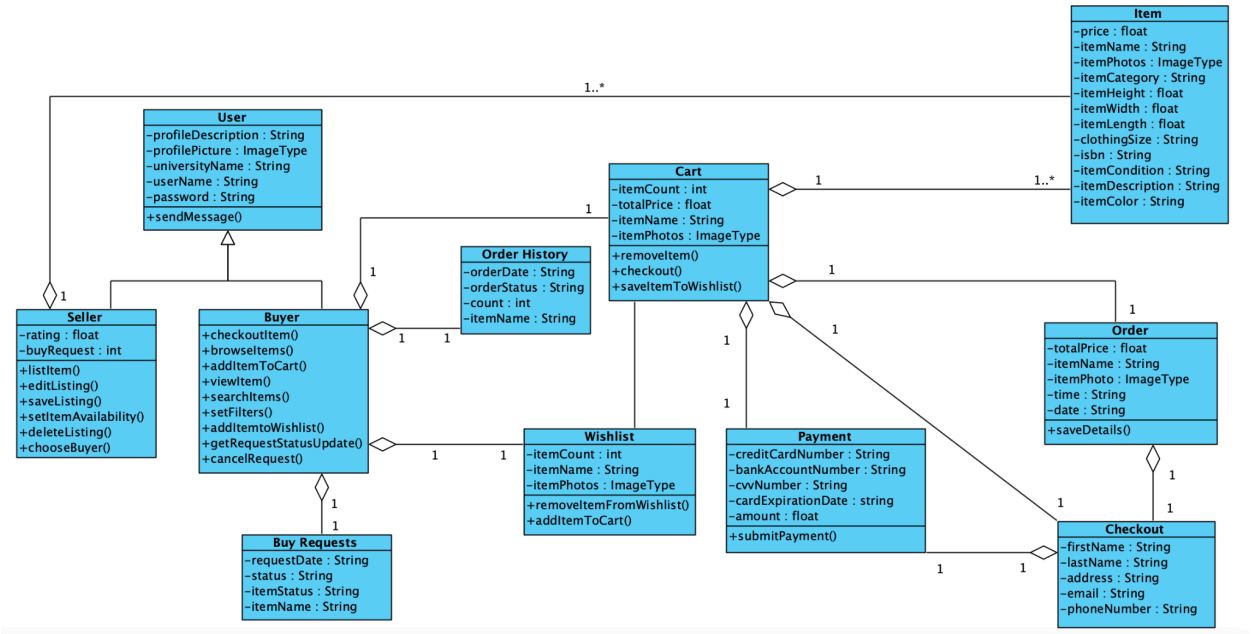
Updates include the added functionality that users are now able to see previous orders which adds a new use case for the Exhcnage4Students platform. Users would now be able to log in to see and interact with previous orders.

Updated Activity Diagram



The updated activity diagram includes the new functionality of checkout confirmation and viewing the order history. Buyers are now able to see there order history and also will be given a confirmation before placing a new order.

Updated Class Diagram



Updates made to our class diagram include the new Buy Requests and Order History entities. The Buy Requests class represents the requests sent to a seller when a buyer checks out an item. Each Buy Request has a requestDate which consists of the time and date the request was sent. It also shows the status of the request (i.e. if the seller accepts, rejects, or has not responded yet) and the status of the item of interest (on hold, available). A single Buy Request is sent for each item a buyer wishes to purchase. Buyers also have the ability to cancel requests if the seller has yet to accept, which is represented by the cancelRequest() operation in the Buyer class. Each buyer also has an Order History where they can view the items they purchased, the date of the order, the total number of orders, and the status of each order.

Design Reflection

A part of our system where thoughtful design was applied was for our digital marketplace and its searching/filtering capabilities. Initially, our marketplace automatically updated with a newly posted listing, did not have search functionality, and displayed posted items in a simple, unresponsive manner. Through our development milestones, we began to grapple with the ideal look and feel of our marketplace, and made improvements such as visually defining its container and the addition of multiple search filters to optimize the user experience. As a team, we learned that even if we believe a design for any component of our app is finalized, the software design process is one that requires constant evolution, so we should anticipate the need for change in our design. As for observed design patterns, our BrowseItems.jsx component contains an indirect behavioral observer pattern for its cart functionality, where React's useEffect listens for updates to the user's cart, and when the cart is modified, the function onCartUpdate notifies a parent component. While not a direct implementation, this component still utilizes the observer pattern by expecting change and notifying other classes when it occurs.