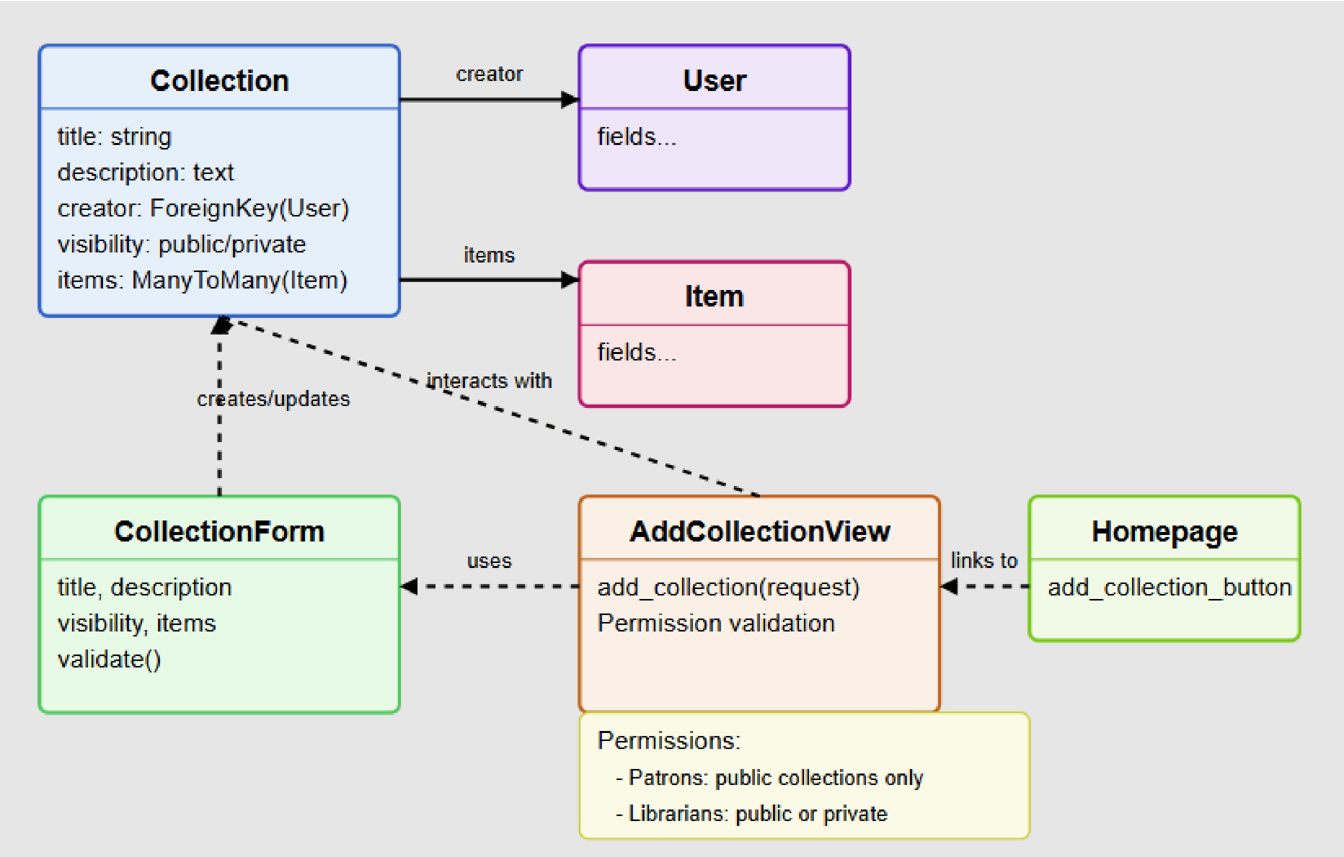
CS 3240 - Requirements Change Analysis **Computing ID: zqh6jp / Team: A-28**

Requirements Change Effects

The requirement change allowing patrons to create collections introduced significant architectural challenges. While the absence of preexisting collection functionality eliminated legacy code conflicts, it demanded a full implementation from scratch, requiring substantial time and effort.

Key Technical Changes:

We created a new Collection model with fields for title, description, items (many-to-many relationship with Item), creator (foreign key to User), and visibility (public/private). Also, we had to make a new collection form in the forms.py file for the new method in views.py file called add_collection. This interacts with the Collection model and validates user permissions (patrons can only create public collections, librarians can do both). In addition, we changed the homepage.html file to include an add_collection_button, which allows users to create a collection from the homepage.



Updates to your own requirements:

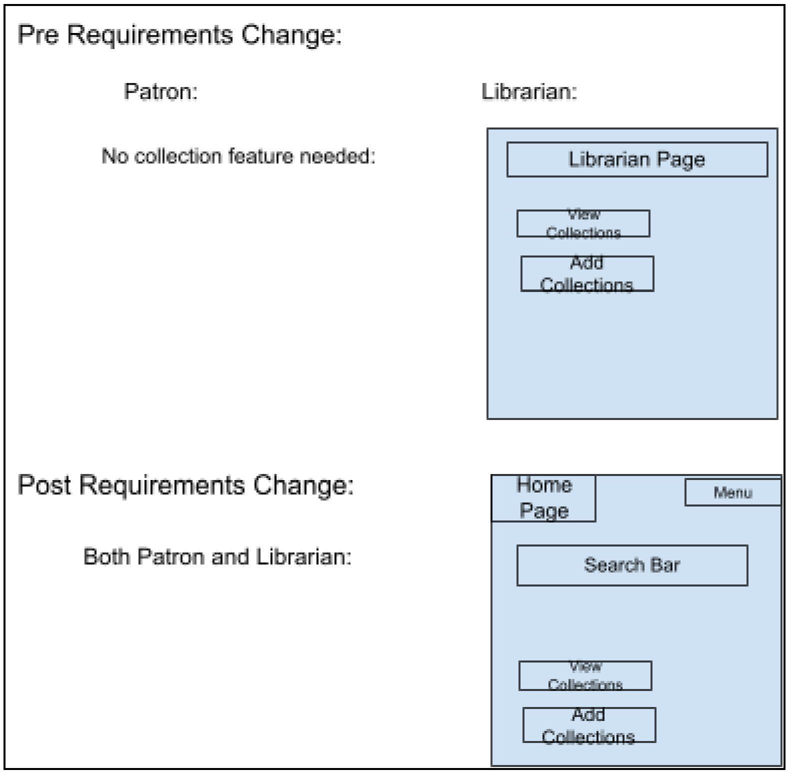
Prior to the requirement change, our focus was on resolving GitHub issues, adding features such as allowing librarians to assign required materials for classes, and fixing bugs related to Google sign-in and profile picture handling. When the new requirement emerged, we realized that the core feature was missing. While this eliminated the need to modify existing code, it introduced the challenge of building the feature from scratch. The changes to the requirements were adding that we had to implement the entire collection, which entails implementing a model for it, adding a way to add collections for users, having a private and public collection, while making it so that items in private collections were not in any other collections. We were also focused on implementing other features and fixing bugs.

Where and how it adds complexity to your project:

Adding the collection feature as a whole causes a lot of complexity issues. It adds complexity to our databases since it uses a new model, deciding how collections would integrate with the items, and where we would visualize and add new collections also adds complexity to the project. The requirements changes cause even more complexity with the collections since we would have to let patrons be able to add only public collections and make sure items that are in private collections can not appear in public collections. These small changes add more layers to the collection model itself. Also, we will need to test the collection feature.

Requirements Change Plan:
In previous sprints, we were mostly independent in our work on our features, with the Scrum Master giving us what features needed to be implemented by the Sprint deadline. This was mostly the same this sprint but the requirement change was mostly with me and the Scrum Master, while the other people were focused on their other features that didn't require the collections. We reassigned it so that two people worked on the requirement change aspect while the other three worked on new independent features. This way we get progress on the overall app while having people dedicated to the requirements change aspect of the Sprint.

The requirements change did pose a challenge for implementing the collection as it isn't as simple as it once was. Before the change we were planning on adding a collection button on a page the librarian can only access to make new collections and bunch them together, and the patron wouldn't have any changes; however, after talking to team members, we decided to put it on the homepage of both the librarian and the patron so they can both add the collections.



Any other changes through the system would be, changing other features to make the UI look better like the marketplace page and the login page, also fully implementing the Required Materials page making Librarians able to add required materials for classes.

In Conclusion:

The requirement change significantly impacted both technical implementation and team coordination. By restructuring task assignments, enforcing validation, and prioritizing testing, we successfully adapted to the new requirements. Future sprints will focus on performance optimizations and addressing technical debt resulting from the accelerated development process.