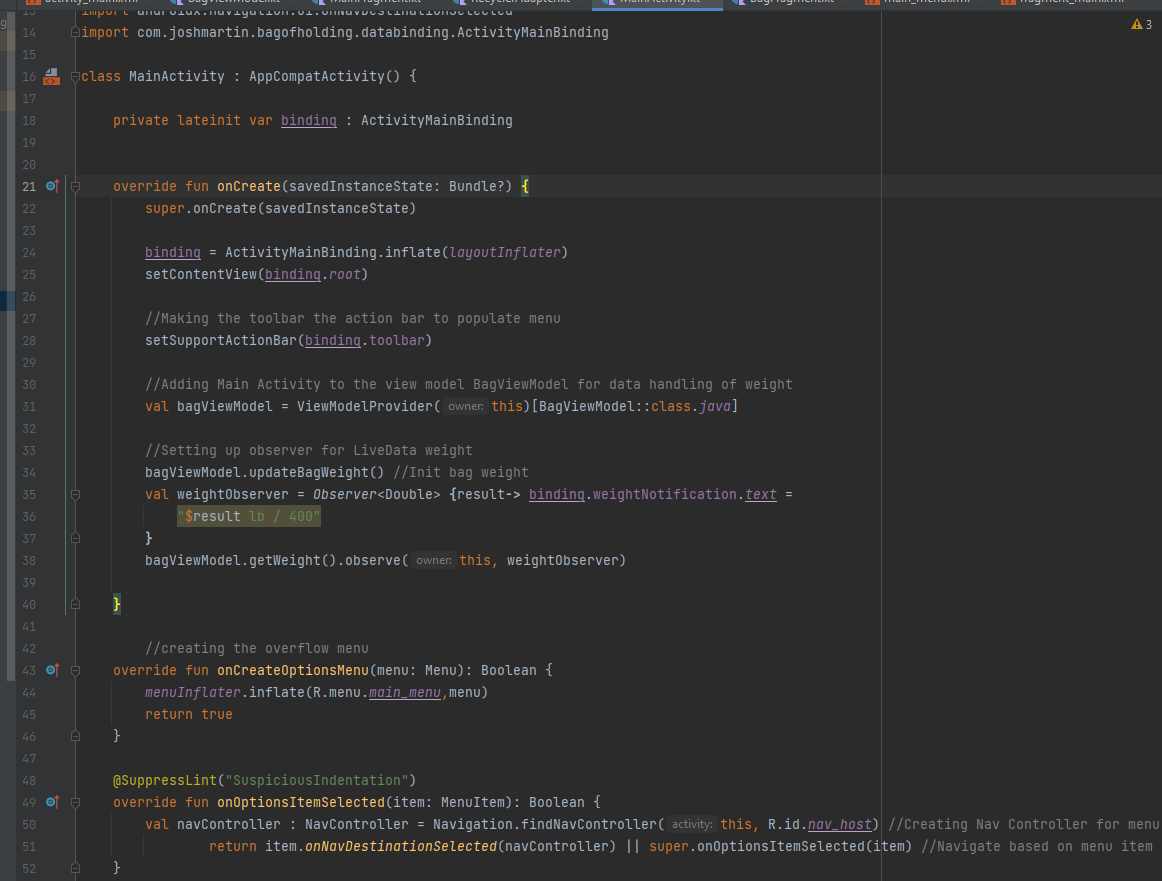
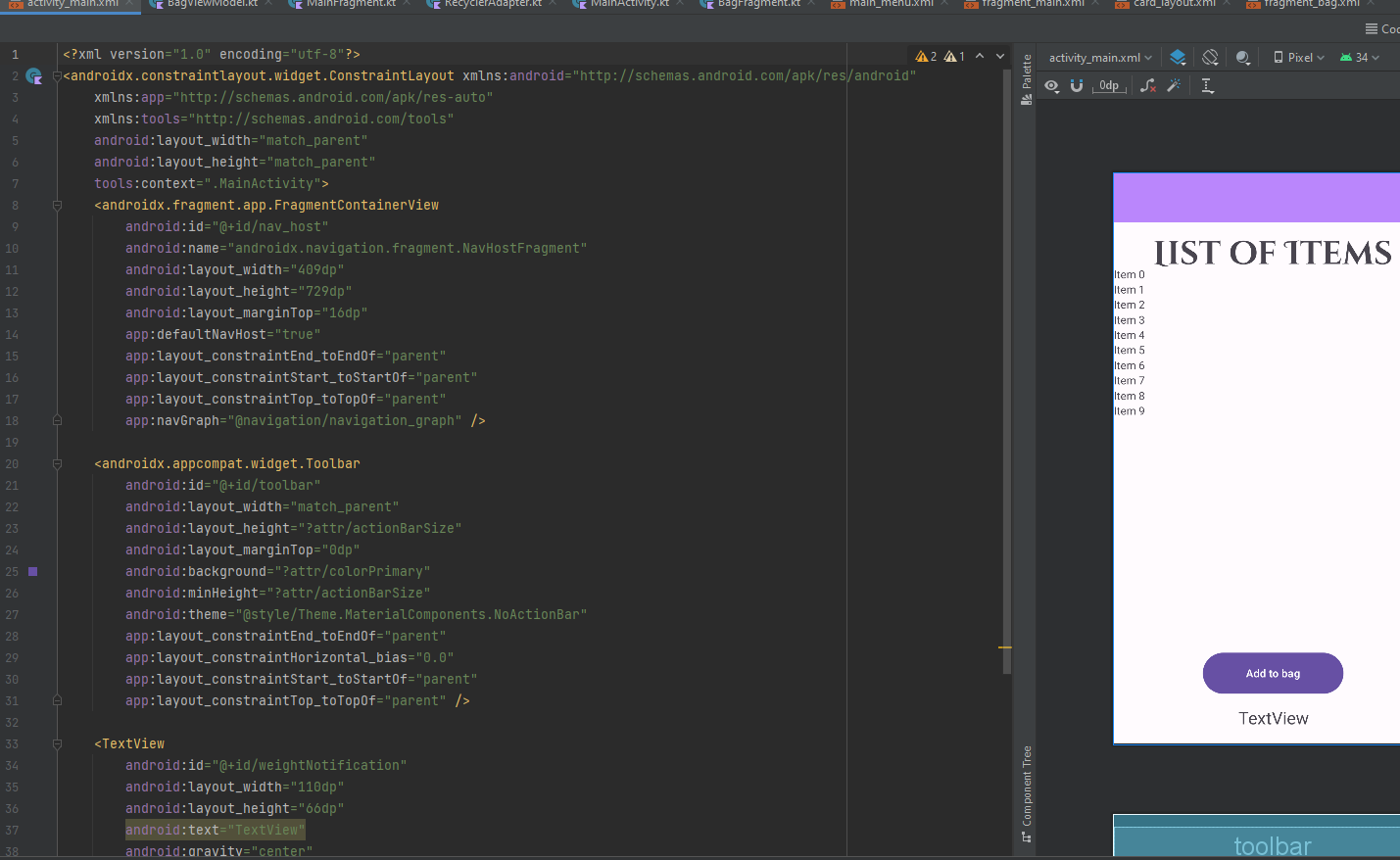
**The Bag of Holding Application**

The Bag of Holding app is a simple application that uses a restful API from open source 5e to generate a list of items that can be added to a bag. The user can then visit their bag and remove any items they wish from it. Bags of Holding are an archetypal D&D element that can store large amounts of items in a small space up to a maximum weight of 400 pounds. I thought this would be a useful app for those of us who play D&D and don’t want to write out every element within their bags of holding on a sheet of paper or in an excel spreadsheet.  
  
**The Main Activity:**

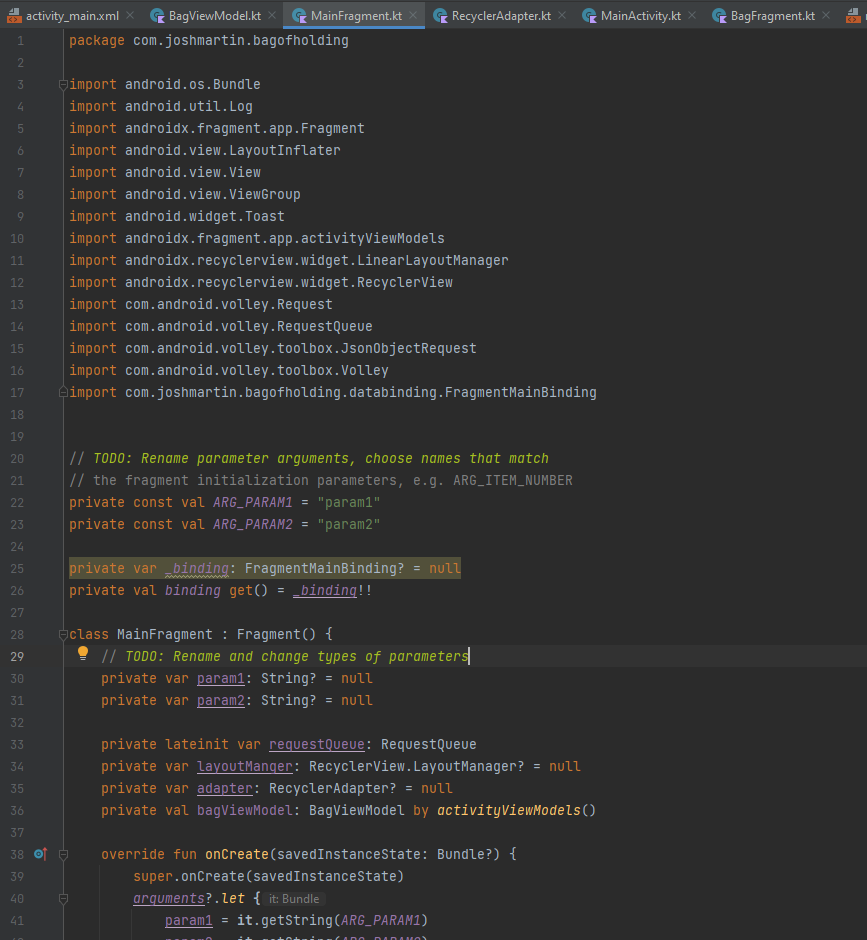
In this activity I create the toolbar and the overflow menu to handle the navigation between my two fragments. I thought that including this in the main activity would be a good idea as the toolbar will not change position from one fragment to another. This involved creating a menu XML for the overflow menu as well as a navigational graph for my navigation. Additionally, I have a text view that displays the total weight of the bag at the bottom of the layout. As with the menu toolbar, I wanted that to be consistent throughout the application. The data for that text view is primarily handled by a bagViewModel class which includes interacting with live data, so that it is always displaying up to date information.

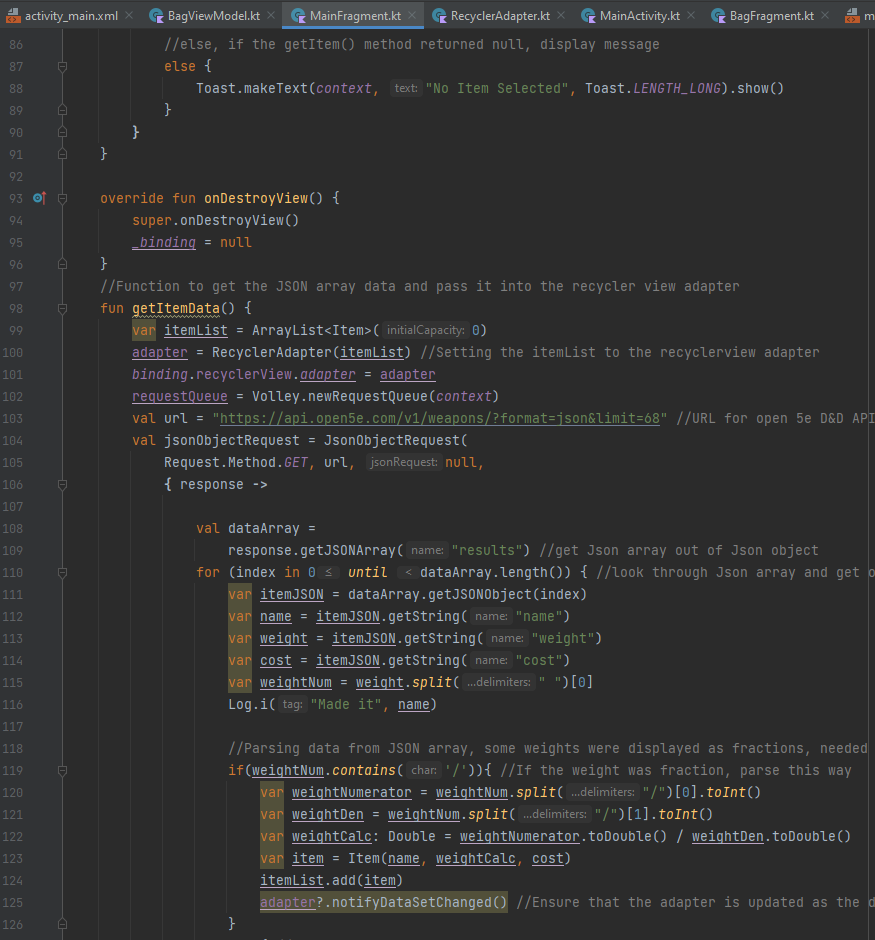
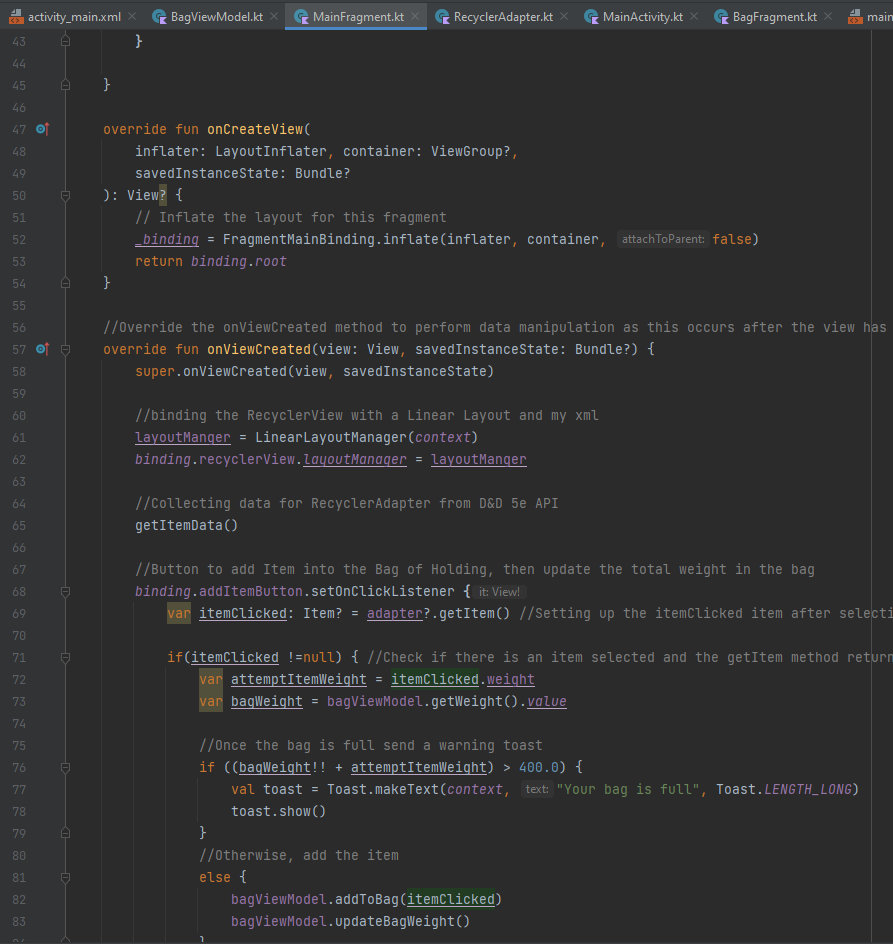
***MainActivity XML***

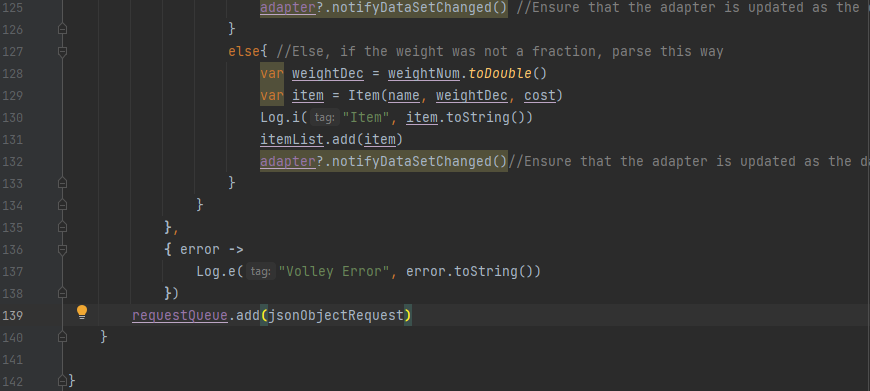
******

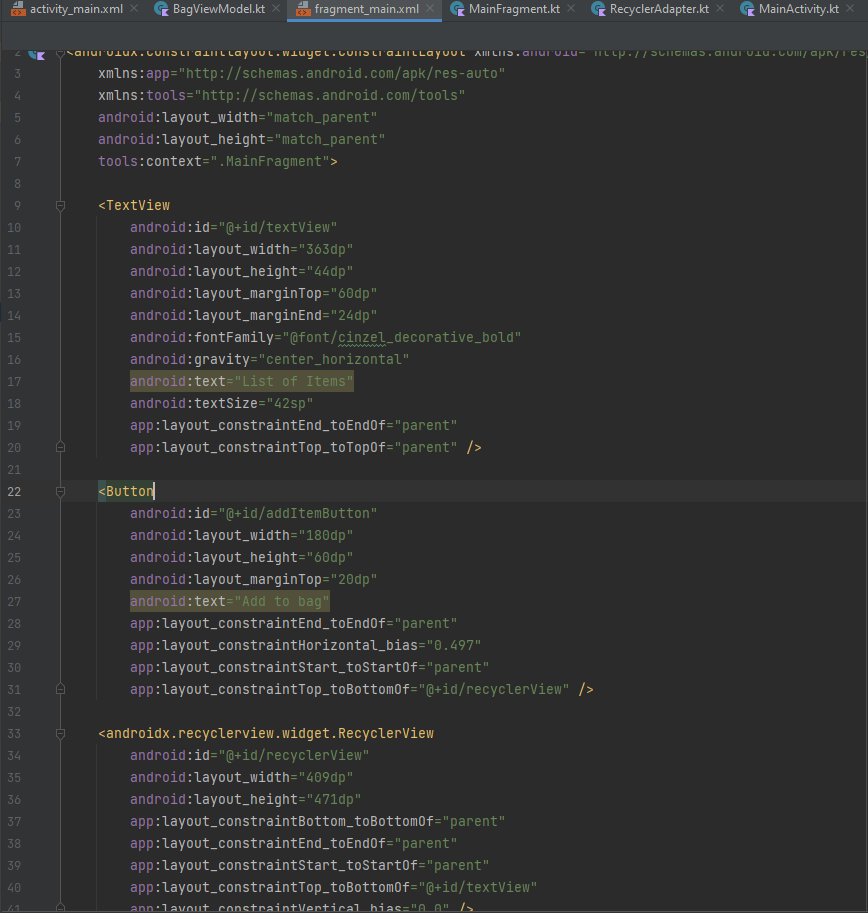
**MainFragment Class**

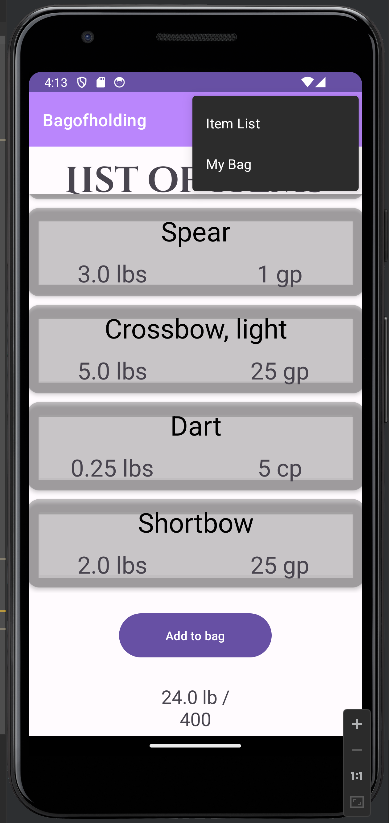
The MainFragment class is responsible for retrieving objects from the open 5e restful API, parsing it and applying it to a custom Cardview/RecyclerView layout that can be interacted with by the user. This involved creating a custom card\_layout xml as well as a custom adapter. The user can click on one of the item cards, which will display information like its name, its weight, and its value, and then click on an “Add Button” to add it to their Bag of Holding. After the user clicks on the “Add Button” a copy of the object is sent to the bagViewModel class and into an arraylist that will be used in the bagFragment. Additionally, each time an item is added to the bag, the bag’s weight is updated, again with the data being processed in bagViewModel.





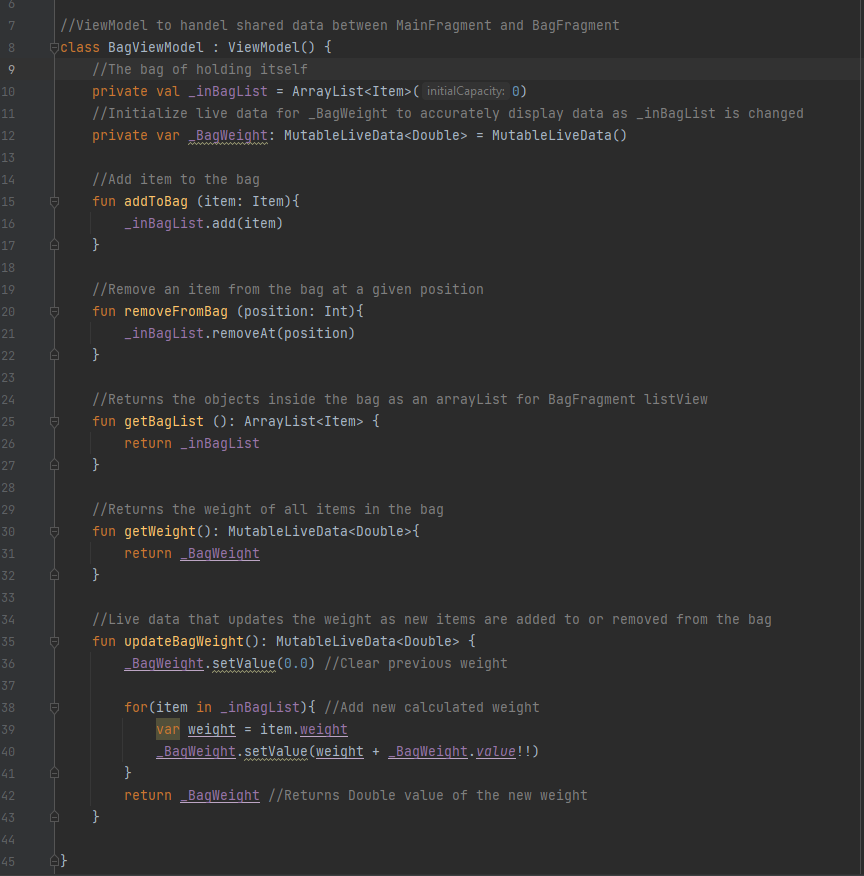


***Main Fragment XML***

**Application Running at the MainFragment:**

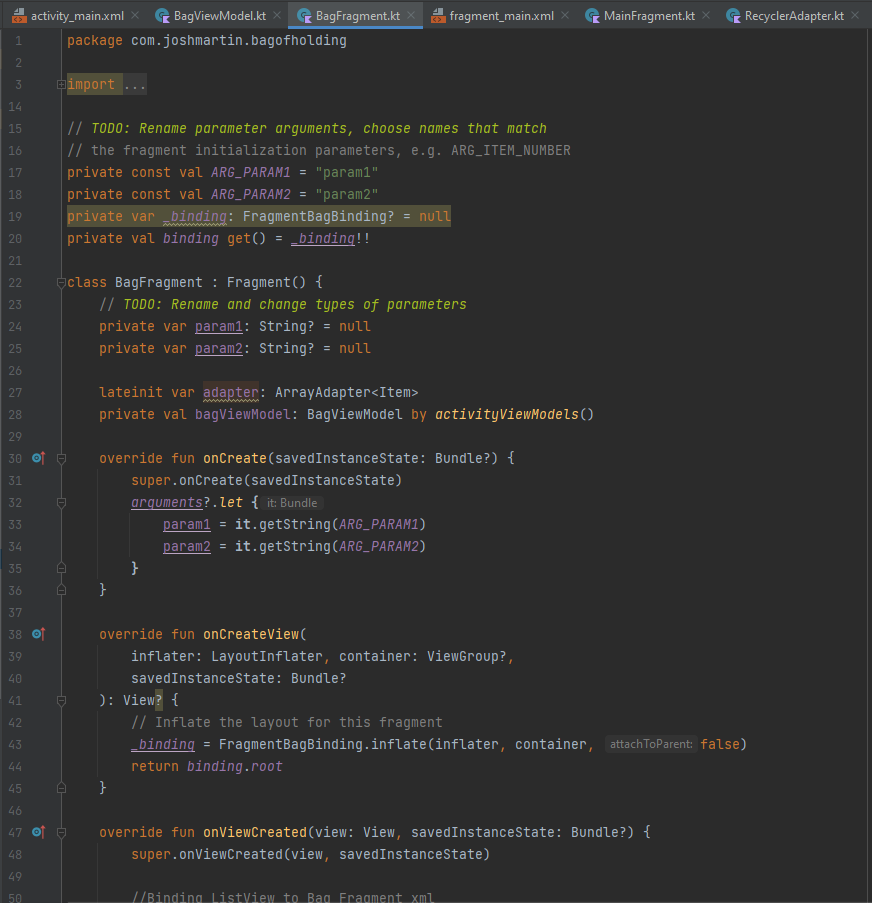
**bagViewModel class:**

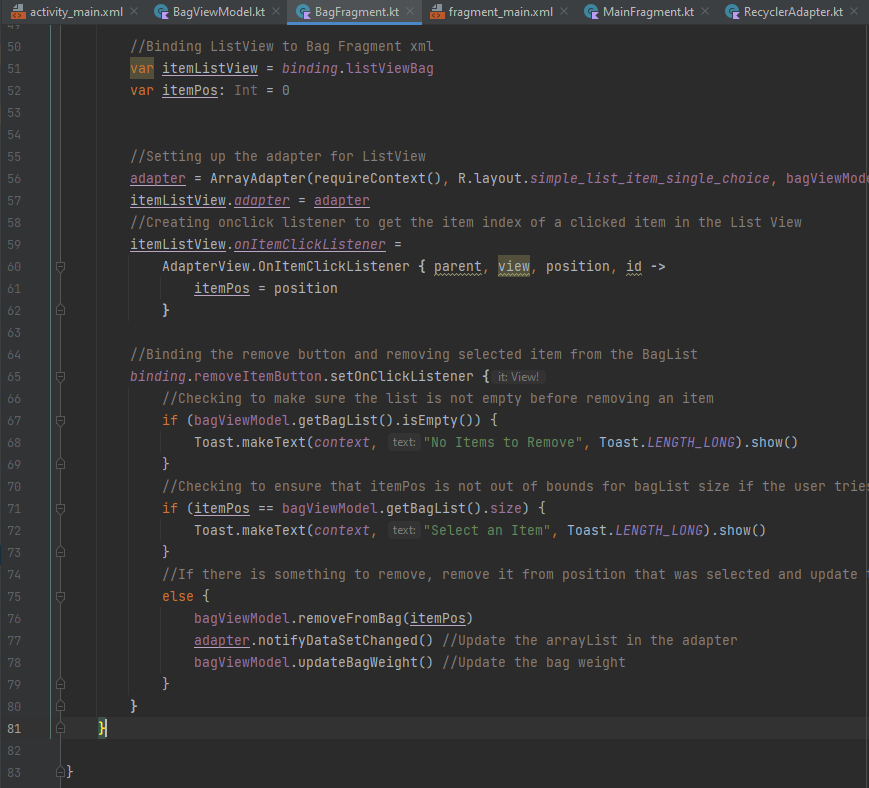
This class is the view model for the project. Its primary responsibility is to perform data processing for the weight of the Bag of Holding as well as a few custom functions in order to add and remove items form an array list that homes all of the items a user has added to the bag.

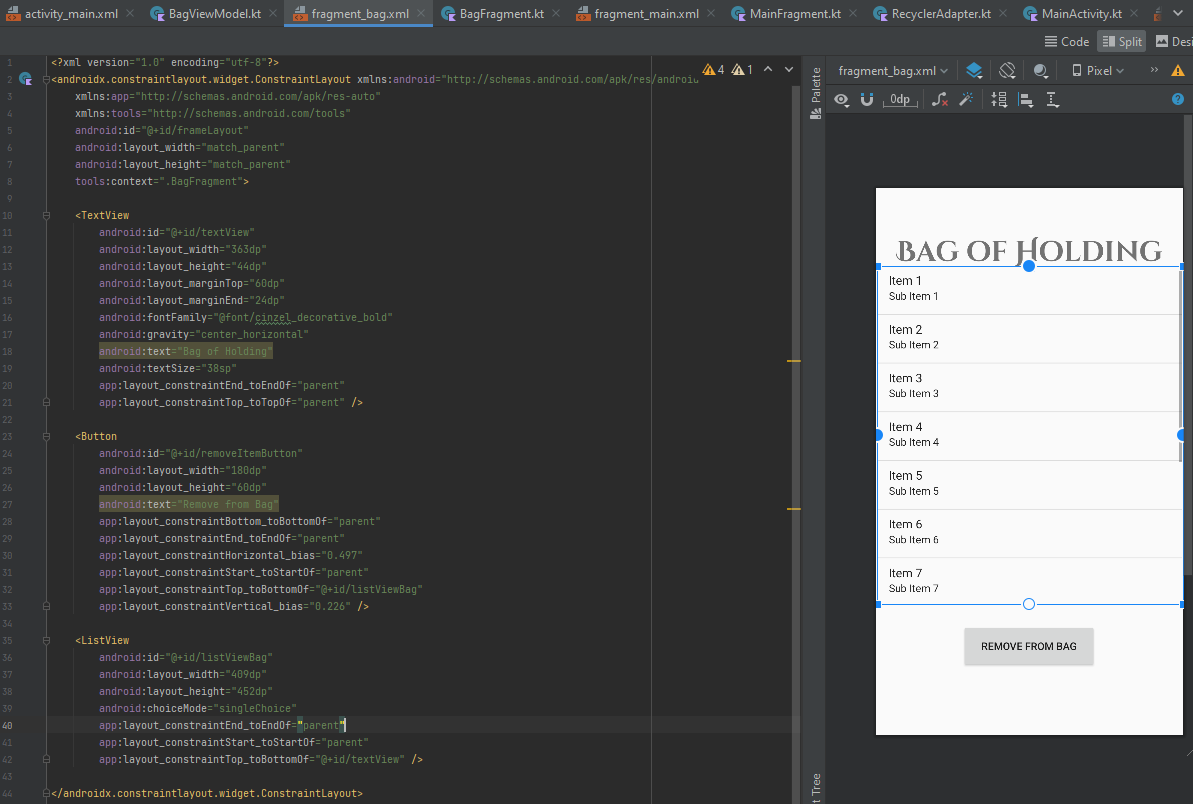


**BagFragment class**

This fragment is used to display the data that the user has added into their bag of holding from the MainFragment. The objects are displayed in a List View with a check box so the user can easily see what item is selected. If they wish, they can remove any item from the list by clicking on the “Remove from Bag” button at the bottom of the layout. The bagViewModel takes care of weight calculations as this occurs as well as updates the \_inBagList which holds the objects within the bag of holding itself.





***bagFragment XML***

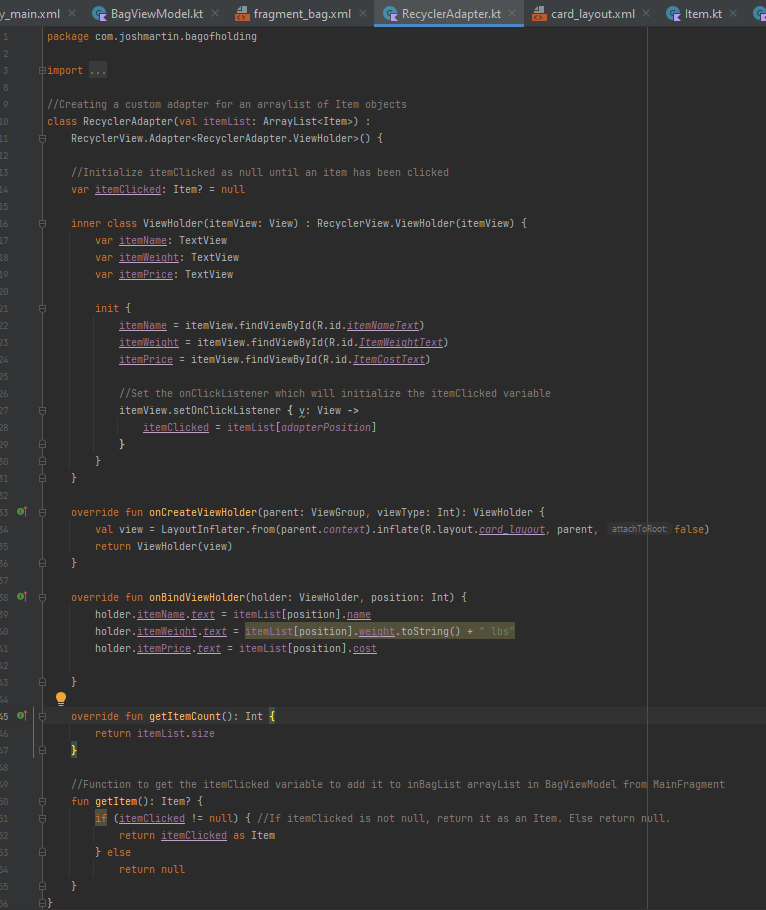
**Application running at the bagFragment:**

**A screenshot of a phone

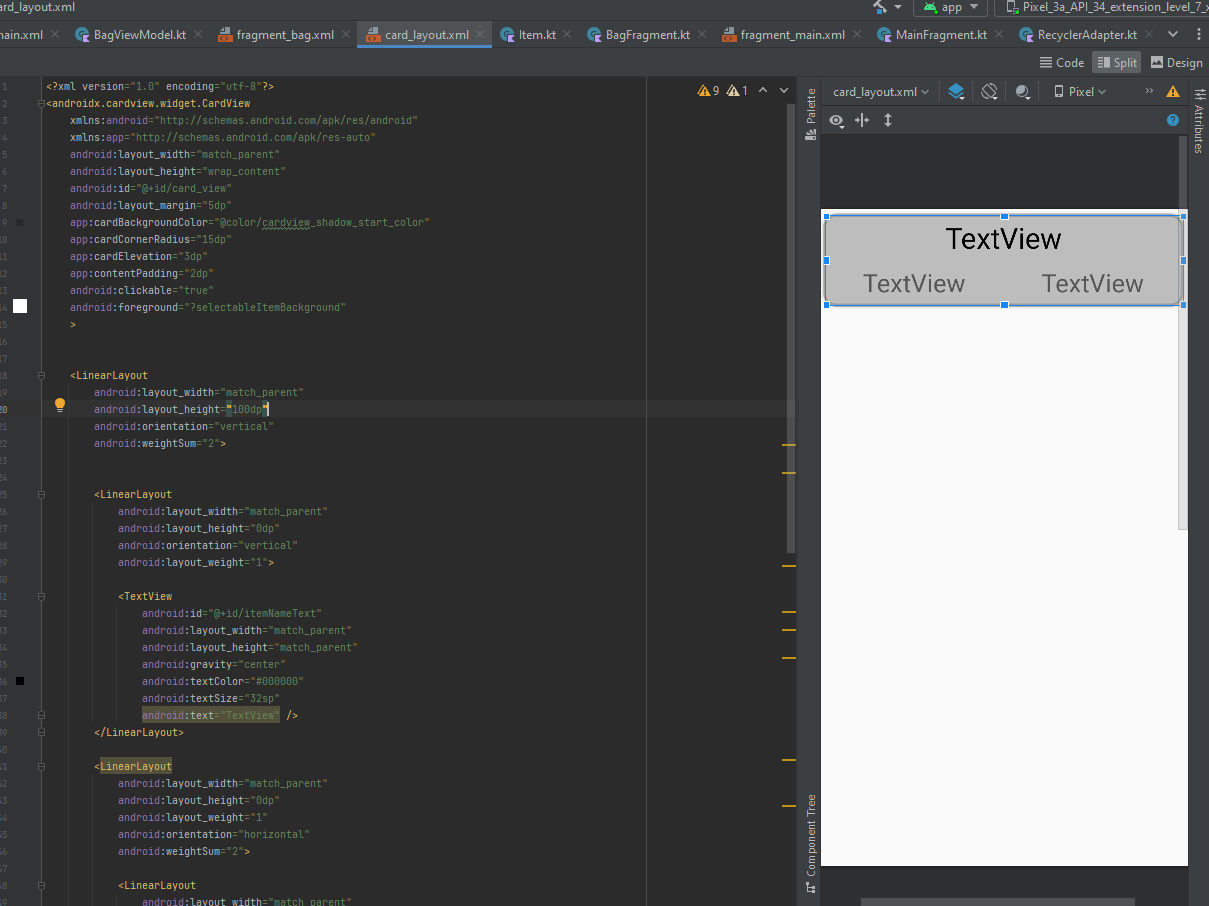
Description automatically generatedA screen shot of a phone

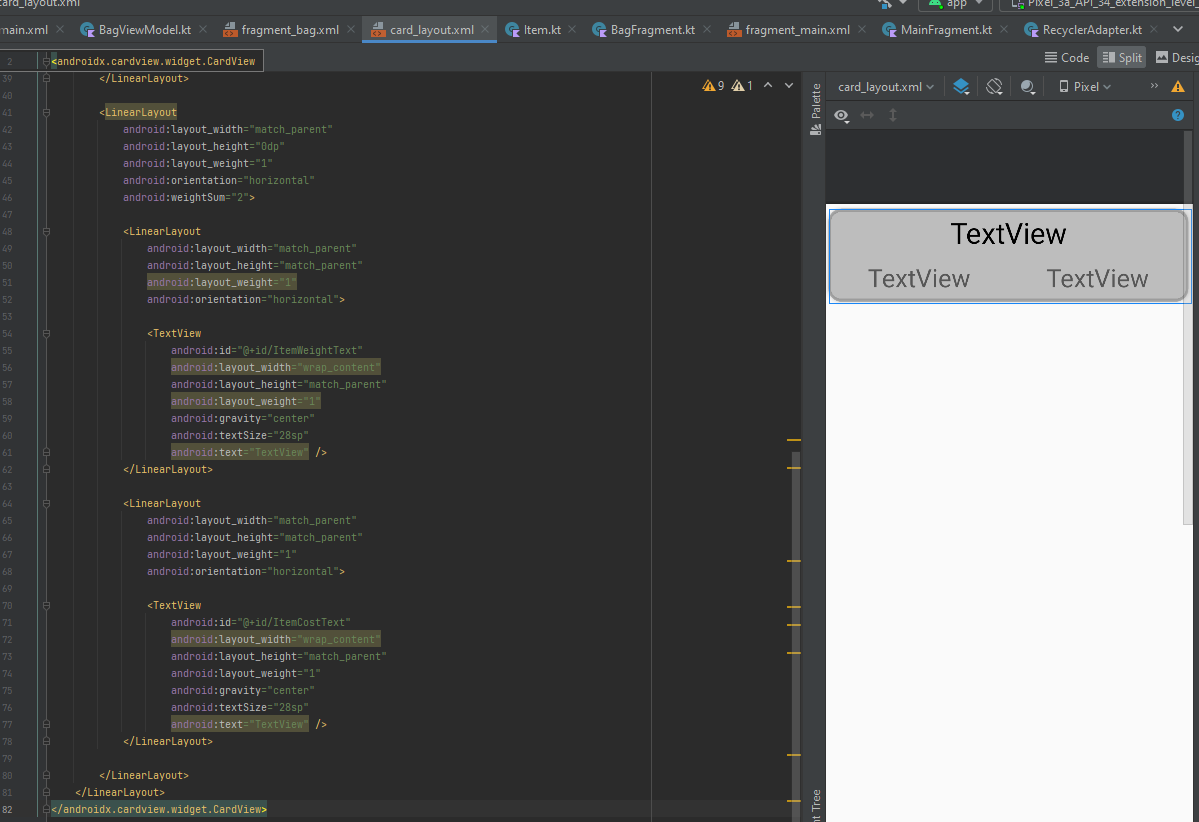
Description automatically generated**

**RecyclerAdapter class**

This is a custom adapter class for my recycler view. It takes in an array list of type Item that is displayed in a card view with the help of my custom card layout. The object data is bound to this card view and the recycler view will cycle through the array list as the user scrolls through it. I also added a setOnClickcedListener event that will store the data of an object that is clicked and then store that into a local object variable. A simple getter function was also added in order to get the object a user clicked on.

***Card\_Layout XML***

This is the xml that is used to define the card layout for my recycler view. It is a linear layout that has been divided in order to display the name of the object at the top, and the weight and value of the object at the bottom.



**Item data class**

A screen shot of a computer

Description automatically generatedThis is a simple custom data class to store the item object data which is used throughout the project.