Stringified Basket:

# Introduction:

A stringified basket is needed because if the user was to exit the tab and then come back, they can continue their shopping without having to add their previous items to the basket again. However, the only exception here is that the user must be on the same device. Otherwise, this will not work.

# Stringified Basket Processes:

We first must understand how the basket feature is implemented. The basket is a 2-Dimensional list. It is a list of items where the items are a list of the item data. An example is shown below in pseudocode:

[

[“basketItemID”, “prodID”, “size”, “quantity”, “name”, “price”],

…

]

To make the basket data stringified, we must apply the String function to each list within the list. This means that we need to iterate through the basket list. An example is shown below:

String([basketItemID, prodID, size, quantity, name, price])

The result of this function is shown below:

“<basketItemID>,<prodID>,<size>,<quantity>,<name>,<price>”

Where the identifier is replaced by the value.

# Separating each item in the stringified basket:

The only thing left that’s needed is to separate items within the stringified basket. We have conveniently chosen the “&” operator. An example of a completely stringified basket is shown below:

"<basketItemID>,<prodID>,<size>,<quantity>,<name>,<price>&"<basketItemID>,<prodID>,<size>,<quantity>,<name>,<price>&"<basketItemID>,<prodID>,<size>,<quantity>,<name>,<price>"

Where the identified is replaced by the value.

# De-stringified baskets:

We must also need de-stringified baskets so that we can load up the user’s basket in an acceptable format. We do the opposite process of the stringified basket process. First, we split by the “&” operator so that we get a list of stringified items. We then iterate through the list of stringified items and split by the “,”. This will result in the item data being reverted back into list form and then we can append the data to the basket list.