Insights:

1) The priority is to get as many people to use the app as possible. Lower amount bills tend to earn lesser coins, which might now prove to be motivating enough for the user to take enough efforts to upload and scan. Walmart as a store has the highest amount in total, and also has the most number of bills uploaded. However, **the cost per bill for Walmart is very less**. This shows that a lot of the offers that Fetch offers are for Walmart stores in general, motivating people to scan the bills despite not being of huge amounts.

2) Few stores like **Burlington, have only a few receipts but they are all of higher amounts**. Fetch can partner with these companies, to get more people to scan bills from these companies. This would help generate more data about the people visiting the stores, and thus help in designing offers.

3) Lastly, **the items sold by Starbucks(irrespective of the store) sum up to the most amount**. Fetch can focus closely on the most frequently bought items, items usually brought together with the Starbucks items and then devise offers, and giftcards for items sold at Starbucks.

Code:

```
%load_ext autoreload
%autoreload 2
%matplotlib inline
```

```
!unzip /content/Takehome_Data_January_2023.zip
```

```
Archive:  /content/Takehome_Data_January_2023.zip
  inflating: users.csv
  inflating: __MACOSX/._users.csv
  inflating: brands.csv
  inflating: __MACOSX/._brands.csv
  inflating: receipt_items.csv
  inflating: __MACOSX/._receipt_items.csv
  inflating: receipts.csv
  inflating: __MACOSX/._receipts.csv
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=Tr
```

```
import pandas as pd
import numpy as np
```

```
brands = pd.read_csv('brands.csv')
receipt_items = pd.read_csv('receipt_items.csv')
receipts = pd.read_csv('receipts.csv')
users = pd.read_csv('users.csv')
```

```
receipts1 = receipts.groupby(by=['STORE_NAME'])['TOTAL_SPENT'].sum()
```

```
receipts1 = pd.DataFrame(receipts1).reset_index()
receipts1.sort_values(by = 'TOTAL_SPENT', ascending = False, inplace = True)
receipts1.reset_index(drop = True, inplace = True)
receipts1.head(20)
```

	STORE_NAME	TOTAL_SPENT
0	WALMART	383000.05
1	BURLINGTON	158594.05
2	THE HOME DEPOT	144646.99
3	COSTCO	131248.32
4	380 LENOX MEAT CORPORATION	125721.85
5	AMAZON	105638.49
6	SAM'S CLUB	95306.62
7	TARGET	84588.84
8	CHASE	68176.92
9	LOWE'S HOME IMPROVEMENT	65731.87

```
receipts2 = pd.DataFrame(receipts['STORE_NAME'].value_counts())
```

```
receipts2.reset_index(inplace = True)
receipts2.columns = ['STORE_NAME', 'RECEIPT_COUNT']
receipts2.head(20)
```

	STORE_NAME	RECEIPT_COUNT
0	WALMART	6931
1	AMAZON	2778
2	TARGET	1805
3	DUNKIN DONUTS	1780
4	MCDONALD'S	1632
5	DOLLAR TREE STORES INC	1403
6	WALGREENS	1393
7	PUBLIX	1224
8	COSTCO	1102
9	SAM'S CLUB	993
10	ALDI	869
11	CVS	854
12	THE HOME DEPOT	845
13	DOLLAR GENERAL STORE	832
14	PRICE CUTTER	683
15	KROGER	678
16	FOOD LION	610
17	STAR DRAGON	551
18	LOWE'S HOME IMPROVEMENT	523
19	KEY FOOD FRESH	513

```
receipts_final = pd.merge(receipts1, receipts2, on=['STORE_NAME'])
receipts_final['AVERAGE_PRICE_PER_BILL'] = receipts_final['TOTAL_SPENT']/receipts_final['RECEIPT_COUNT']
receipts_final1 = receipts_final.sort_values(by = 'AVERAGE_PRICE_PER_BILL', ascending = False).reset_index(drop = True)
receipts_final2 = receipts_final1[receipts_final1['RECEIPT_COUNT'] > 10]
receipts_final2[receipts_final2['TOTAL_SPENT'] > 0]
```

	STORE_NAME	TOTAL_SPENT	RECEIPT_COUNT	AVERAGE_PRICE_PER_BILL
8	BURLINGTON	158594.05	73	2172.521233
20	380 LENOX MEAT CORPORATION	125721.85	100	1257.218500
22	PIONEER SUPERMARKETS	63038.93	61	1033.425082
52	KOHLS	8570.68	15	571.378667
117	BEST BUY	21639.96	68	318.234706
...	...	...	...	...
5626	CROSSROADS CAFE	98.94	51	1.940000

```
items1 = receipt_items.groupby(by=['BRAND_CODE'])['TOTAL_FINAL_PRICE'].sum()
items1 = pd.DataFrame(items1).reset_index()
items1.sort_values(by = 'TOTAL_FINAL_PRICE', ascending = False, inplace = True)
items1.reset_index(drop = True, inplace = True)
items1.head(20)
```

	BRAND_CODE	TOTAL_FINAL_PRICE
0	STARBUCKS	64715.82
1	FRESH	62568.76
2	KIRKLAND SIGNATURE	22221.71
3	GREAT VALUE	19669.38
4	MEMBER'S MARK	11303.68
5	KROGER	7041.33
6	PUBLIX	6574.47
7	MARLBORO	6183.76
8	GE	5629.94
9	APPLE	5403.16
10	H-E-B	5348.47
11	COKE	5187.09
12	SCOTT	5047.47
13	PEPSI	4781.71
14	LIFEPROOF	4571.18
15	COORS LIGHT	4339.98
16	GATORADE	4178.97
17	COSTCO	3939.63
18	MAINSTAYS	3607.87
19	DEWALT	3542.83

✓ 0s completed at 3:58 PM