Проект исследования поведения пользователей в мобильном приложении ¶

Цели проекта

- Изучить поведение пользователей мобильного приложения по продаже продуктов питания;
- Построить воронки продаж и посмотреть какой путь проходят пользователи и где могут быть точки роста у приложения;
- Также необходимо провести A/A и A/B тесты проверяя как влияет изменения шрифта приложения имея 2 контрольные выборки и 1 новую выборку.

Описание данных

EventName — название события;
DeviceIDHash — ID пользователя;
EventTimestamp — время события;
ExpId — номер груп для тестов: 246 и 247 — контрольные группы, а 248 — экспериментальная.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import datetime as dt
import seaborn as sns
import plotly.express as px
from plotly import graph_objects as go
import numpy as np
from datetime import datetime
from scipy import stats as st
import math as mth
import cmath
import nbconvert
from IPython.display import Javascript
from nbconvert import HTMLExporter
```

Шаг 1

In [2]:
```
df = pd.read_csv('logs_exp.csv', sep='\t')
df = df.rename(columns={"EventName": "event", "DeviceIDHash": "device_id", "EventTimestamp": "time_stamp", "ExpId":"test_id"})
# переименовал названия столбцов для удобства

display(df.head(10))
```

	event	device_id	time_stamp	test_id
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246
2	PaymentScreenSuccessful	3518123091307005509	1564054127	248
3	CartScreenAppear	3518123091307005509	1564054127	248
4	PaymentScreenSuccessful	6217807653094995999	1564055322	248
5	CartScreenAppear	6217807653094995999	1564055323	248
6	OffersScreenAppear	8351860793733343758	1564066242	246
7	MainScreenAppear	5682100281902512875	1564085677	246
8	MainScreenAppear	1850981295691852772	1564086702	247
9	MainScreenAppear	5407636962369102641	1564112112	246

In [3]:
```
display(df.info())
display(df.isnull().sum())
df = df.drop_duplicates()
#пропусков нет, ушло немного дубликатов, так как всего из было около тысячи для такого массива данных это ок
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244126 entries, 0 to 244125
Data columns (total 4 columns):
 #   Column      Non-Null Count   Dtype 
---  ------      --------------   ----- 
 0   event       244126 non-null  object
 1   device_id   244126 non-null  int64 
 2   time_stamp  244126 non-null  int64 
 3   test_id     244126 non-null  int64 
dtypes: int64(3), object(1)
memory usage: 7.5+ MB


None

event         0
device_id     0
time_stamp    0
test_id       0
dtype: int64
```

In [4]:
```
display(df['event'].value_counts())
display(df['device_id'].value_counts())
display(df['test_id'].value_counts())
```

```
MainScreenAppear           119101
OffersScreenAppear          46808
CartScreenAppear            42668
PaymentScreenSuccessful     34118
Tutorial                     1018
Name: event, dtype: int64

6304868067479728361    2307
19702789326556566O     1998
4623191541214045580    1768
6932517045703054087    1439
1754140665440434215    1221
                       ...
3336727186673646149       1
1657967711232741323       1
425817683219936619        1
4089770943116790924       1
1083512226259476085       1
Name: device_id, Length: 7551, dtype: int64

248    85582
246    80181
247    77950
Name: test_id, dtype: int64
```

In [5]:
```
df['time_stamp'] = pd.to_datetime(df['time_stamp'], unit='s')
df['date'] = df['time_stamp'].dt.date.astype('datetime64')
df['time'] = df['time_stamp'].dt.time#.astype('datetime64')
# перевели нужные столбцы в даты и добавил стобцы только с датой и только с верменем
```

In [6]:
```
display(df['time_stamp'])
display(df.info())
display(df.head())
```

```
0         2019-07-25 04:43:36
1         2019-07-25 11:11:42
2         2019-07-25 11:28:47
3         2019-07-25 11:28:47
4         2019-07-25 11:48:42
                  ...
244121    2019-08-07 21:12:25
244122    2019-08-07 21:13:59
244123    2019-08-07 21:14:43
244124    2019-08-07 21:14:58
244125    2019-08-07 21:15:17
Name: time_stamp, Length: 243713, dtype: datetime64[ns]

<class 'pandas.core.frame.DataFrame'>
Int64Index: 243713 entries, 0 to 244125
Data columns (total 6 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   event       243713 non-null  object
 1   device_id   243713 non-null  int64
 2   time_stamp  243713 non-null  datetime64[ns]
 3   test_id     243713 non-null  int64
 4   date        243713 non-null  datetime64[ns]
 5   time        243713 non-null  object
dtypes: datetime64[ns](2), int64(2), object(2)
memory usage: 13.0+ MB

None
```

	event	device_id	time_stamp	test_id	date	time
0	MainScreenAppear	4575588528974610257	2019-07-25 04:43:36	246	2019-07-25	04:43:36
1	MainScreenAppear	7416695313311560658	2019-07-25 11:11:42	246	2019-07-25	11:11:42
2	PaymentScreenSuccessful	3518123091307005509	2019-07-25 11:28:47	248	2019-07-25	11:28:47
3	CartScreenAppear	3518123091307005509	2019-07-25 11:28:47	248	2019-07-25	11:28:47
4	PaymentScreenSuccessful	6217807653094995999	2019-07-25 11:48:42	248	2019-07-25	11:48:42

Подготовили данные к последующей работеа именно:

- убрал дубликаты, ушло немного около 1к значений
- перевели столбцы в нужные форматы
- переименовал названия столбцов для удобства

- изучил данные

Шаг 2

In [7]:
```
display(df['event'].value_counts()) #все события
display(df['device_id'].nunique()) # количество уникальных пользователей
display(df.groupby('event').agg({"device_id": "nunique"}).sort_values(
    by='device_id', ascending=False)) #смотрим сколько приходится событий на уникальных пользователей
```

```
MainScreenAppear           119101
OffersScreenAppear          46808
CartScreenAppear            42668
PaymentScreenSuccessful     34118
Tutorial                     1018
Name: event, dtype: int64

7551
```

event	device_id
MainScreenAppear	7439
OffersScreenAppear	4613
CartScreenAppear	3749
PaymentScreenSuccessful	3547
Tutorial	847

In [8]:
```
display(df['date'].describe())
#display(df['date'].hist(bins=30))
date_hist = px.histogram(df, x="date",nbins=25,title='распределение действий пользователей по дате',labels={'date':'дата'})
date_hist.show()
```

```
count                  243713
unique                     14
top       2019-08-01 00:00:00
freq                    36141
first     2019-07-25 00:00:00
last      2019-08-07 00:00:00
Name: date, dtype: object
```

Как видно из гистограммы несмотря то что первое дейтсвие у нас было совершенно ещё 25 июля то до августа у практически не было активности, поэтому будем считать от 1 августа включительно, и откинем данные ранее, у нас получится неделя данных

In [9]:
```
display(df.query('date >="2019-08-01"').info())
#если откинем данные до 1 августа, то всего потеряем около 3 тыс, что совсем немного для такого массива
df = df.query('date >="2019-08-01"')
```

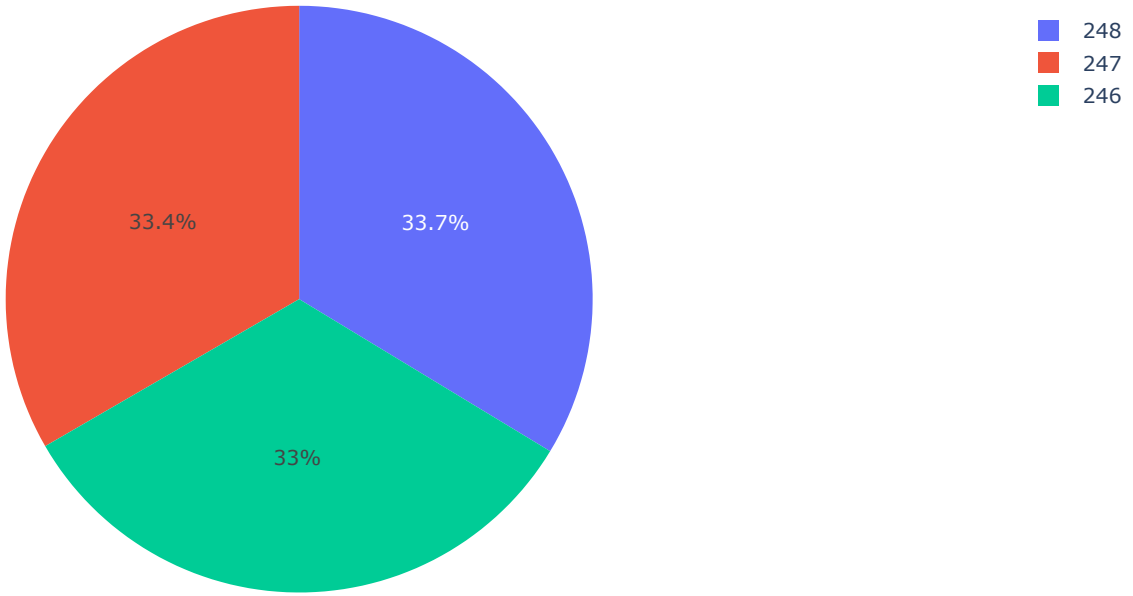
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 240887 entries, 2828 to 244125
Data columns (total 6 columns):
 #   Column      Non-Null Count   Dtype         
---  ------      --------------   -----         
 0   event       240887 non-null  object        
 1   device_id   240887 non-null  int64         
 2   time_stamp  240887 non-null  datetime64[ns]
 3   test_id     240887 non-null  int64         
 4   date        240887 non-null  datetime64[ns]
 5   time        240887 non-null  object        
dtypes: datetime64[ns](2), int64(2), object(2)
memory usage: 12.9+ MB

None
```

In [10]:
```
display(df['test_id'].value_counts())
# также есть все выборки експерементальные, всё ок
```

```
248    84563
246    79302
247    77022
Name: test_id, dtype: int64
```

In [11]:
```
display(df.groupby('test_id').agg({"device_id": "nunique"}).sort_values(
    by='device_id', ascending=False))
# количество уникальных пользователей в каждой из групп


#
fig = go.Figure(data=[go.Pie(labels=['246','247','248'],
                             values=[2484,2513,2537])])
fig.show()
```

test_id	device_id
248	2537
247	2513
246	2484



Определили временную выборку с 1 августа, так как до 1 августа было мало действий. По сути получили недельную выборку активности пользователей Групы также распределенны верно, разница в выборках меньше 1%, что показывает нам корректность разделения на группы

Шаг 3

In [12]:
```
display(df.groupby('event').agg({"device_id": "nunique"}).sort_values(
    by='device_id', ascending=False))
```

event	device_id
MainScreenAppear	7419
OffersScreenAppear	4593
CartScreenAppear	3734
PaymentScreenSuccessful	3539
Tutorial	840

Судя по групировки мы видим что наш пользователе проходит путь: главная страничка - страничка с продуктами - страничка с выбором оплаты и других данных - окно удачной оплаты. Есть ещё окно "помощь", но он не в общей цепочки действий, а точнее скорее всего не всегда в ней. интересно будет посмотреть детальней как это окно влияет на пользователей

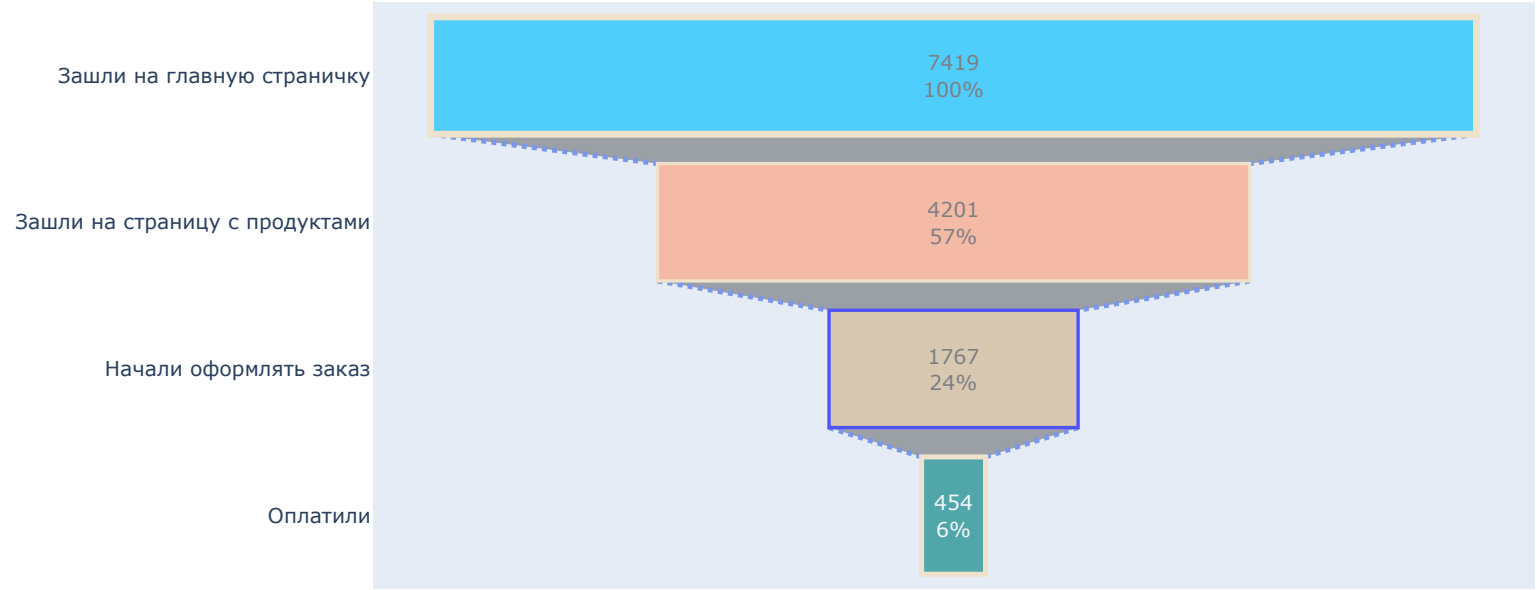
In [13]:
```
users = df.pivot_table(
        index='device_id',
        columns='event',
        values='time_stamp',
        aggfunc='min')

display(users.head(15))
```

event	CartScreenAppear	MainScreenAppear	OffersScreenAppear	PaymentScreenSuccessful	Tutorial
device_id					
6888746892508752	NaT	2019-08-06 14:06:34	NaT	NaT	NaT
6909561520679493	2019-08-06 18:52:58	2019-08-06 18:52:54	2019-08-06 18:53:04	2019-08-06 18:52:58	NaT
6922444491712477	2019-08-04 14:19:40	2019-08-04 14:19:33	2019-08-04 14:19:46	2019-08-04 14:19:40	NaT
7435777799948366	NaT	2019-08-05 08:06:34	NaT	NaT	NaT
7702139951469979	2019-08-02 14:28:45	2019-08-01 04:29:54	2019-08-01 04:29:56	2019-08-02 14:28:45	NaT
8486814028069281	2019-08-05 04:49:18	2019-08-05 04:52:40	2019-08-05 04:49:13	NaT	NaT
8740973466195562	NaT	2019-08-02 09:16:48	2019-08-02 09:43:59	NaT	NaT
9841258664663090	2019-08-03 10:52:15	2019-08-03 10:47:59	2019-08-03 10:49:42	2019-08-03 17:57:27	2019-08-03 10:47:28
12692216027168046	NaT	2019-08-02 16:28:49	2019-08-05 04:06:02	NaT	NaT
15708180189885246	2019-08-01 11:06:19	2019-08-01 16:08:23	2019-08-01 05:38:55	2019-08-01 11:06:19	NaT
18658818197810381	2019-08-01 15:25:32	2019-08-02 03:05:37	2019-08-01 08:00:42	2019-08-03 13:47:35	NaT
20449203507642281	NaT	2019-08-06 21:04:21	NaT	NaT	NaT
20795828045873027	2019-08-01 08:01:55	2019-08-01 08:01:47	2019-08-01 08:01:48	2019-08-01 08:01:55	NaT
24144092107925848	NaT	2019-08-01 12:53:16	NaT	NaT	NaT
26317307137967461	NaT	2019-08-01 14:33:57	NaT	NaT	NaT

In [14]:
```
# посмтроим воронку с учётом порядка действий пользователей

step_1 = ~users['MainScreenAppear'].isna()
step_2 = step_1 & (users['OffersScreenAppear'] > users['MainScreenAppear'])
step_3 = step_2 & (users['CartScreenAppear'] > users['OffersScreenAppear'])
step_4 = step_3 & (users['PaymentScreenSuccessful'] > users['CartScreenAppear'])
step_5 = step_4 & (users['CartScreenAppear'] > users['Tutorial'])

n_main_page = users[step_1].shape[0]
n_offer = users[step_2].shape[0]
n_cart = users[step_3].shape[0]
n_payment = users[step_4].shape[0]
n_tutorial = users[step_5].shape[0]

print('Зашли на главную страничку:', n_main_page)
print('Зашли на страницу с продуктами:', n_offer)
print('Начали оформлять заказ:', n_cart)
print('Оплатили:', n_payment)
print('Посмотрели справку',n_tutorial )
```

```
Зашли на главную страничку: 7419
Зашли на страницу с продуктами: 4201
Начали оформлять заказ: 1767
Оплатили: 454
Посмотрели справку 217
```

In [15]:
```
fig = go.Figure(go.Funnel(
    y = ["Зашли на главную страничку", "Зашли на страницу с продуктами", "Начали оформлять заказ", "Оплатили"],
    x = [7419, 4201, 1767, 454],
    textposition = "inside",
    textinfo = "value+percent initial",
    opacity = 0.65, marker = {"color": ["deepskyblue", "lightsalmon", "tan", "teal", "silver"],
    "line": {"width": [4, 2, 2, 3, 1, 1], "color": ["wheat", "wheat", "blue", "wheat", "wheat"]}},
    connector = {"line": {"color": "royalblue", "dash": "dot", "width": 3}})
    )

fig.show()
```



Визуализация воронки показывает показывает следующие тенденции:

- 57% занитересовались нашим сервисом и перешли с главной страницы на страницу с выборобом продуктов
- 42% от тех кто просмотрел продукты начали оформлять заказ

- всего 25% от тех кто начал оформлять заказл, удачно его оплатили
- 47% тех кто просмотрели справку удачно оплатили заказ, а всего справку посмотрело всего около 3% из всех польщователей

Иходя из полученных результатов я бы предварительно выдвинул такие гипотезы и реккомендации:

- нужно сделать более видимую страницу с справкой , так половина успешно оплаченных заказов, проходили через окно справки, что показывает важность этого показателя для финальной оплаты;
- большой разрыв между главной страницой и страницой выбора товара. Думаю либо нужно улучшать основную страницу приложение, либо же сразу выводить на выбор товара что бы сократить время перехода;
- видимо есть проблемы с оформлением заказа, так как всего 25% начали и закончили оформлять заказ, а у других пользователей видно не получилось. Может сюда стоит втроить более простую схему оформления заказа и справку где-то держать рядом, так как она положительно влияет на конверсию

In [16]:
```
df_246 = df.query('test_id == "246"')
df_247 = df.query('test_id == "247"')
df_248 = df.query('test_id == "248"')
```

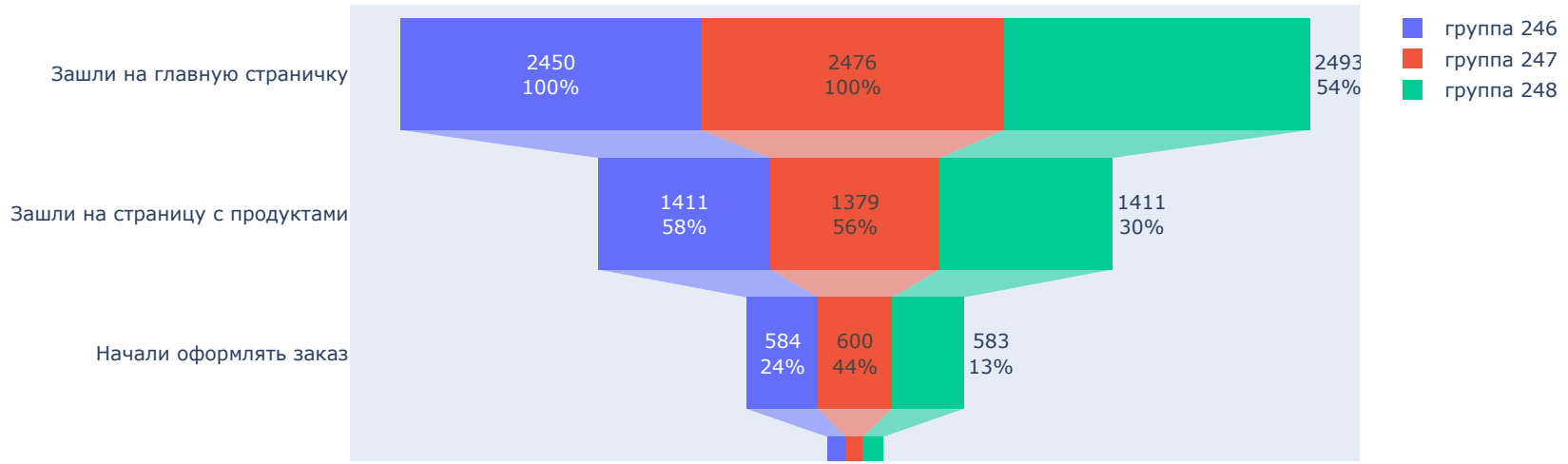
In [17]:
```
# делаю для графика funnel
users_246 = df_246.pivot_table(
        index='device_id',
        columns='event',
        values='time_stamp',
        aggfunc='min')

users_247 = df_247.pivot_table(
        index='device_id',
        columns='event',
        values='time_stamp',
        aggfunc='min')

users_248 = df_248.pivot_table(
        index='device_id',
        columns='event',
        values='time_stamp',
        aggfunc='min')
```

In [18]:
```
# 246
step_1_246 = ~users_246['MainScreenAppear'].isna()
step_2_246 = step_1_246 & (users_246['OffersScreenAppear'] > users_246['MainScreenAppear'])
step_3_246 = step_2_246 & (users_246['CartScreenAppear'] > users_246['OffersScreenAppear'])
step_4_246 = step_3_246 & (users_246['PaymentScreenSuccessful'] > users_246['CartScreenAppear'])
step_5_246 = step_4_246 & (users_246['CartScreenAppear'] > users_246['Tutorial'])

n_main_page_246 = users_246[step_1_246].shape[0]
n_offer_246 = users_246[step_2_246].shape[0]
n_cart_246 = users_246[step_3_246].shape[0]
n_payment_246 = users_246[step_4_246].shape[0]
n_tutorial_246 = users_246[step_5_246].shape[0]

print('Зашли на главную страничку:', n_main_page_246)
print('Зашли на страницу с продуктами:', n_offer_246)
print('Начали оформлять заказ:', n_cart_246)
print('Оплатили:', n_payment_246)
print('Посмотрели справку',n_tutorial_246)

# 247
step_1_247 = ~users_247['MainScreenAppear'].isna()
step_2_247 = step_1_247 & (users_247['OffersScreenAppear'] > users_247['MainScreenAppear'])
step_3_247 = step_2_247 & (users_247['CartScreenAppear'] > users_247['OffersScreenAppear'])
step_4_247 = step_3_247 & (users_247['PaymentScreenSuccessful'] > users_247['CartScreenAppear'])
step_5_247 = step_4_247 & (users_247['CartScreenAppear'] > users_247['Tutorial'])

n_main_page_247 = users_247[step_1_247].shape[0]
n_offer_247 = users_247[step_2_247].shape[0]
n_cart_247 = users_247[step_3_247].shape[0]
n_payment_247 = users_247[step_4_247].shape[0]
n_tutorial_247 = users_247[step_5_247].shape[0]

print('Зашли на главную страничку:', n_main_page_247)
print('Зашли на страницу с продуктами:', n_offer_247)
print('Начали оформлять заказ:', n_cart_247)
print('Оплатили:', n_payment_247)
print('Посмотрели справку',n_tutorial_247)

# 248

step_1_248 = ~users_248['MainScreenAppear'].isna()
step_2_248 = step_1_248 & (users_248['OffersScreenAppear'] > users_248['MainScreenAppear'])
step_3_248 = step_2_248 & (users_248['CartScreenAppear'] > users_248['OffersScreenAppear'])
step_4_248 = step_3_248 & (users_248['PaymentScreenSuccessful'] > users_248['CartScreenAppear'])
step_5_248 = step_4_248 & (users_248['CartScreenAppear'] > users_248['Tutorial'])

n_main_page_248 = users_248[step_1_248].shape[0]
n_offer_248 = users_248[step_2_248].shape[0]
n_cart_248 = users_248[step_3_248].shape[0]
```

```
n_payment_248 = users_248[step_4_248].shape[0]
n_tutorial_248 = users_248[step_5_248].shape[0]

print('Зашли на главную страничку:', n_main_page_248)
print('Зашли на страницу с продуктами:', n_offer_248)
print('Начали оформлять заказ:', n_cart_248)
print('Оплатили:', n_payment_248)
print('Посмотрели справку',n_tutorial_248)
```

```
Зашли на главную страничку: 2450
Зашли на страницу с продуктами: 1411
Начали оформлять заказ: 584
Оплатили: 145
Посмотрели справку 72
Зашли на главную страничку: 2476
Зашли на страницу с продуктами: 1379
Начали оформлять заказ: 600
Оплатили: 144
Посмотрели справку 63
Зашли на главную страничку: 2493
Зашли на страницу с продуктами: 1411
Начали оформлять заказ: 583
Оплатили: 165
Посмотрели справку 82
```

In [19]:
```
fig = go.Figure()

fig.add_trace(go.Funnel(
    name = 'группа 246',
    y = ["Зашли на главную страничку", "Зашли на страницу с продуктами", "Начали оформлять заказ", "Оплатили"],
    x = [2450, 1411, 584, 145],
    textinfo = "value+percent initial"))

fig.add_trace(go.Funnel(
    name = 'группа 247',
    orientation = "h",
    y = ["Зашли на главную страничку", "Зашли на страницу с продуктами", "Начали оформлять заказ", "Оплатили"],
    x = [2476, 1379, 600, 144],
    textposition = "inside",
    textinfo = "value+percent previous"))

fig.add_trace(go.Funnel(
    name = 'группа 248',
    orientation = "h",
    y = ["Зашли на главную страничку", "Зашли на страницу с продуктами", "Начали оформлять заказ", "Оплатили"],
    x = [2493, 1411, 583, 165],
    textposition = "outside",
    textinfo = "value+percent total"))

fig.show()
```



Также для наглядности постролили воронку по разбивке кажой группы

Исходя воронки разбитой по контрольным группам и новой наша конверсия в этих группах более менне равна для всех случаев, но вот у новой группы конверсия в оплату всё таки немного лучше а именно 6,6% от изначальных (против 5.9% и 5.8%) ну и выше по переходу оформление - успешанпя оплата

Шаг 4

Проводим A/A тест

так как у нас нормальное распределние и мы будем проверять гипотезы о равентсве нашей пропорции среди групп 246 и 247 будем испольщовать z test Фишера Нулевая гипотеза: между долями груп нет стастистической разницы Альтернативаня гипотеза: отвергаем нулевую и есть основания считать, что между долями груп есть разница

In [20]:
```
# даелаем функцию для проверки гипотез

def a_a_test(data1, data2): #где data1 это количество просмотров от предидущего шага, а data2 количество перезодов на след. уровень
    alpha = .1 # критический уровень статистической значимости

    p1 = data2[0] / data1[0]
    p2 = data2[1] / data1[1]
    p_combined = (data2[0] + data2[1]) / (data1[0] + data1[1])
    difference = p1 - p2

    z_value = difference / cmath.sqrt(p_combined * (1 - p_combined) * (1/data1[0] + 1/data1[1]))

    distr = st.norm(0,1)

    p_value = (1 - distr.cdf(abs(z_value))) * 2

    print('p-значение: ', p_value)

    if (p_value < alpha):
        print("Отвергаем нулевую гипотезу: между долями есть значимая разница")
    else:
        print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными")
```

проверяем различие долей для перехода главная страница - страница с продуктами

In [21]:
```
data2 = n_offer_246, n_offer_247
data1 = n_main_page_246, n_main_page_247
display(a_a_test(data1, data2))
```

```
p-значение:  0.1791392160319234
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с продуктами - оформление заказа

In [22]:
```
data2 = n_cart_246, n_cart_247
data1 = n_offer_246, n_offer_247
display(a_a_test(data1, data2))
```

```
p-значение:  0.2571602924470018
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с оформление заказа - перход к оплате

In [23]:
```
data2 = n_payment_246, n_payment_247
data1 = n_cart_246, n_cart_247
display(a_a_test(data1, data2))
```

```
p-значение:  0.7399516568824782
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода главаня страница - перход к справке

In [24]:
```
data2 = n_tutorial_246, n_tutorial_247
data1 = n_main_page_246, n_main_page_247
display(a_a_test(data1, data2))
```

```
p-значение:  0.39664401954186523
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода оформление заказа - перход к справке

In [25]:
```
data2 = n_tutorial_246, n_tutorial_247
data1 = n_cart_246, n_cart_247
display(a_a_test(data1, data2))
```

```
p-значение:  0.3222560323282291
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

Наш A/A анализ показал что во всех тестах нет оснований отвергать нулевую гипотезу, и между нашими долями нет статистически значимой разницы. Это говорит нам о том что с нашим разбиением на группы 246 и 247 всё нормально, и то что наше разбиение на группы сработало корректно

Переходим к A/B тестам

Для корректного продолжения експеримента будем продолжать использовать z критерий Фишера. Определим такие гипотезы:

- Нулевая гипотеза: между долями нет статистически значимой разницы

- Альтернативаня гипотеза: есть основания считать, что между долями есть значимая разница

A/B тест груп 266 и 268

проверяем различие долей для перехода главная страница - страница с продуктами

In [26]:
```
data2 = n_offer_246, n_offer_248
data1 = n_main_page_246, n_main_page_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.48049838808343814
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с продуктами - оформление заказа

In [27]:
```
data2 = n_cart_246, n_cart_248
data1 = n_offer_246, n_offer_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.9695085280919828
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с оформление заказа - перход к оплате

In [28]:
```
data2 = n_payment_246, n_payment_248
data1 = n_cart_246, n_cart_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.17922330576112433
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода главаня страница - перход к справке

In [29]:
```
data2 = n_tutorial_246, n_tutorial_248
data1 = n_main_page_246, n_main_page_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.47830720155927886
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода оформление заказа - перход к справке

In [30]:
```
data2 = n_tutorial_246, n_tutorial_248
data1 = n_cart_246, n_cart_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.3808546194879363
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

после проведения A/B тестов между контрольной группой 266 и новой 268 можно сказать, что статистически значимой разницы между долями нет, как при 0.1 так и при 0.05 уровню стастистической значимости

A/B тест груп 267 и 268

проверяем различие долей для перехода главная страница - страница с продуктами

In [31]:
```
data2 = n_offer_247, n_offer_248
data1 = n_main_page_247, n_main_page_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.5208915431808141
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с продуктами - оформление заказа

In [32]:
```
data2 = n_cart_247, n_cart_248
data1 = n_offer_247, n_offer_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.24154783250462764
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с оформление заказа - перход к оплате

In [33]:
```
data2 = n_payment_247, n_payment_248
data1 = n_cart_247, n_cart_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.09219350245330471
Отвергаем нулевую гипотезу: между долями есть значимая разница

None
```

проверяем различие долей для перехода главаня страница - перход к справке

In [34]:
```
data2 = n_tutorial_247, n_tutorial_248
data1 = n_main_page_247, n_main_page_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.11885443090833969
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода оформление заказа - перход к справке

In [35]:
```
data2 = n_tutorial_247, n_tutorial_248
data1 = n_cart_247, n_cart_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.061567883096199205
Отвергаем нулевую гипотезу: между долями есть значимая разница

None
```

после A/B тестов при критическом уровне 0.05 статистически значимых заличий в контрольной выборке и новой нет, но вот при уровне 0.1 тест показывает нам что есть разница в долях при переходе со страницы оформления заказа - справка и оформление заказа - оплата

A/B тест груп общей группы (266+267) и 268

проверяем различие долей для перехода главная страница - страница с продуктами в общей выборке и новой

In [36]:
```
data2 = n_offer_247+n_offer_246, n_offer_248
data1 = n_main_page_247+n_main_page_246, n_main_page_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.9739544491299794
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с продуктами - оформление заказа

In [37]:
```
data2 = n_cart_247+n_cart_246, n_cart_248
data1 = n_offer_247+n_offer_246, n_offer_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.48772425017293974
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода страница с оформление заказа - перход к оплате

In [38]:
```
data2 = n_payment_247+n_payment_246, n_payment_248
data1 = n_cart_247+n_cart_246, n_cart_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.07823520976058496
Отвергаем нулевую гипотезу: между долями есть значимая разница

None
```

проверяем различие долей для перехода главаня страница - перход к справке общей контрольной и новой

In [39]:
```
data2 = n_tutorial_247+n_tutorial_246, n_tutorial_248
data1 = n_main_page_247+n_main_page_246, n_main_page_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.18526859775696214
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

проверяем различие долей для перехода оформление - перход к справке общей контрольной и новой

In [40]:
```
data2 = n_tutorial_247+n_tutorial_246, n_tutorial_248
data1 = n_cart_247+n_cart_246, n_cart_248
display(a_a_test(data1, data2))
```

```
p-значение:  0.10877556383306075
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

None
```

после A/B тестов при критическом уровне 0.05 статистически значимых заличий в контрольной выборке и новой нет, но вот при уровне 0.1 тест показывает нам что есть разница в долях при переходе со страницы оформления заказа - оплата

во время тестов критический уровень статистической значимости был 0.05 так и 0.1 и в следующих тестах были такие показатели:

- A/A тест: во обоих случаях изменений не было
- A/B тест группа 266 и 268: во обоих случаях изменений не было
- A/B тест группа 267 и 268: при уровне 0.05 изменений не было, а вот при 0.1 были в переходах с оформления - справки и оформление заказа - оплата разница была
- A/B тест группа 267+266 и 268: при уровне 0.05 изменений не было, а вот при 0.1 были в переходах с оформления - оплата есть разница

Всего мы провели в общей сложности 40 тестов, 10 A/A тестов и 30 A/B тестов и только 3 из них показали отличия в выборках, что может говорить о том что они ложные (каждые 10 тестов могут иметь ложный результат), но далее об этом в общих выводах

Общие выводы

В данной работе мы построили как общую продуктовую воронку так и в разбивке по контрольным группам также провели 40 тестов из которых 10 A/A тестов и 30 A/B тестов и можно сделать такой вывод:

- после проведения тестов у нас возникает разница в долях между в оформлении заказов - справка и офрмление заказов - оплата, но так как было 40 тестов можно считать, что эти тесты дали ложный результат. Но, если детальней изучить воронку, то можно заметить, что у новой группы показатели конверсии, практически везде лучше. Исходя из проведенного анализа, я бы сказал, что шрифты менять в этой ситуации можно и оно даже немного улучшает конверсию

In [ ]: