Importing libraries

In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Loading dataset

In [7]:
```
df =pd.read_csv('Hotel_bookings 2.csv')
```

Initial Exploratory Analysis and Cleaning

In [8]:
```
df.head()
```

Out[8]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_
0	Resort Hotel	0	342	2015	July	27	
1	Resort Hotel	0	737	2015	July	27	
2	Resort Hotel	0	7	2015	July	27	
3	Resort Hotel	0	13	2015	July	27	
4	Resort Hotel	0	14	2015	July	27	

5 rows × 32 columns

In [9]:
```
df.tail()
```

Out[9]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_d
119385	City Hotel	0	23	2017	August	35	
119386	City Hotel	0	102	2017	August	35	
119387	City Hotel	0	34	2017	August	35	
119388	City Hotel	0	109	2017	August	35	
119389	City Hotel	0	205	2017	August	35	

5 rows × 32 columns

In [10]:
```
df.shape
```

Out[10]:
```
(119390, 32)
```

In [11]:
```
df.columns
```

Out[11]:
```
Index(['hotel', 'is_canceled', 'lead_time', 'arrival_date_year',
       'arrival_date_month', 'arrival_date_week_number',
       'arrival_date_day_of_month', 'stays_in_weekend_nights',
       'stays_in_week_nights', 'adults', 'children', 'babies', 'meal',
       'country', 'market_segment', 'distribution_channel',
       'is_repeated_guest', 'previous_cancellations',
       'previous_bookings_not_canceled', 'reserved_room_type',
       'assigned_room_type', 'booking_changes', 'deposit_type', 'agent',
       'company', 'days_in_waiting_list', 'customer_type', 'adr',
       'required_car_parking_spaces', 'total_of_special_requests',
       'reservation_status', 'reservation_status_date'],
      dtype='object')
```

In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119390 non-null  object
 1   is_canceled                     119390 non-null  int64
 2   lead_time                       119390 non-null  int64
 3   arrival_date_year               119390 non-null  int64
 4   arrival_date_month              119390 non-null  object
 5   arrival_date_week_number        119390 non-null  int64
 6   arrival_date_day_of_month       119390 non-null  int64
 7   stays_in_weekend_nights         119390 non-null  int64
 8   stays_in_week_nights            119390 non-null  int64
 9   adults                          119390 non-null  int64
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64
 12  meal                            119390 non-null  object
 13  country                         118902 non-null  object
 14  market_segment                  119390 non-null  object
 15  distribution_channel            119390 non-null  object
 16  is_repeated_guest               119390 non-null  int64
 17  previous_cancellations          119390 non-null  int64
 18  previous_bookings_not_canceled  119390 non-null  int64
 19  reserved_room_type              119390 non-null  object
 20  assigned_room_type              119390 non-null  object
 21  booking_changes                 119390 non-null  int64
 22  deposit_type                    119390 non-null  object
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64
 26  customer_type                   119390 non-null  object
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
dtypes: float64(4), int64(16), object(12)
memory usage: 29.1+ MB
```

reservation_status_date needs to be changed from obj dtype to datetime dtype

In [13]:
```
df['reservation_status_date'] = pd.to_datetime(df['reservation_status_date'] , format='%
```

In [14]:
```
print(df['reservation_status_date'].dtype)
```

```
datetime64[ns]
```

there are also obj dtypes, that are the categorical columns, and see the unique values in those columns

In [15]:
```
df.describe(include = object) # if you write 'include' and 'obj'- you'll see the summary
```

Out[15]:

	hotel	arrival_date_month	meal	country	market_segment	distribution_channel	reserved_room_type
count	119390	119390	119390	118902	119390	119390	119390
unique	2	12	5	177	8	5	10
top	City Hotel	August	BB	PRT	Online TA	TA/TO	A
freq	79330	13877	92310	48590	56477	97870	85994

now what are the categories in the object columns (we now know the nubmers) need to run a loop on those columns that have dtype 'object'

In [16]:
```
for column in df.describe(include = object).columns:
    print(column)
    print(df[column].unique())# filtering the df with that column and then
    #passing the 'unique' function
```

```
hotel
['Resort Hotel' 'City Hotel']
arrival_date_month
['July' 'August' 'September' 'October' 'November' 'December' 'January'
 'February' 'March' 'April' 'May' 'June']
meal
['BB' 'FB' 'HB' 'SC' 'Undefined']
country
['PRT' 'GBR' 'USA' 'ESP' 'IRL' 'FRA' nan 'ROU' 'NOR' 'OMN' 'ARG' 'POL'
 'DEU' 'BEL' 'CHE' 'CN' 'GRC' 'ITA' 'NLD' 'DNK' 'RUS' 'SWE' 'AUS' 'EST'
 'CZE' 'BRA' 'FIN' 'MOZ' 'BWA' 'LUX' 'SVN' 'ALB' 'IND' 'CHN' 'MEX' 'MAR'
 'UKR' 'SMR' 'LVA' 'PRI' 'SRB' 'CHL' 'AUT' 'BLR' 'LTU' 'TUR' 'ZAF' 'AGO'
 'ISR' 'CYM' 'ZMB' 'CPV' 'ZWE' 'DZA' 'KOR' 'CRI' 'HUN' 'ARE' 'TUN' 'JAM'
 'HRV' 'HKG' 'IRN' 'GEO' 'AND' 'GIB' 'URY' 'JEY' 'CAF' 'CYP' 'COL' 'GGY'
 'KWT' 'NGA' 'MDV' 'VEN' 'SVK' 'FJI' 'KAZ' 'PAK' 'IDN' 'LBN' 'PHL' 'SEN'
 'SYC' 'AZE' 'BHR' 'NZL' 'THA' 'DOM' 'MKD' 'MYS' 'ARM' 'JPN' 'LKA' 'CUB'
 'CMR' 'BIH' 'MUS' 'COM' 'SUR' 'UGA' 'BGR' 'CIV' 'JOR' 'SYR' 'SGP' 'BDI'
 'SAU' 'VNM' 'PLW' 'QAT' 'EGY' 'PER' 'MLT' 'MWI' 'ECU' 'MDG' 'ISL' 'UZB'
 'NPL' 'BHS' 'MAC' 'TGO' 'TWN' 'DJI' 'STP' 'KNA' 'ETH' 'IRQ' 'HND' 'RWA'
 'KHM' 'MCO' 'BGD' 'IMN' 'TJK' 'NIC' 'BEN' 'VGB' 'TZA' 'GAB' 'GHA' 'TMP'
 'GLP' 'KEN' 'LIE' 'GNB' 'MNE' 'UMI' 'MYT' 'FRO' 'MMR' 'PAN' 'BFA' 'LBY'
 'MLI' 'NAM' 'BOL' 'PRY' 'BRB' 'ABW' 'AIA' 'SLV' 'DMA' 'PYF' 'GUY' 'LCA'
 'ATA' 'GTM' 'ASM' 'MRT' 'NCL' 'KIR' 'SDN' 'ATF' 'SLE' 'LAO']
market_segment
['Direct' 'Corporate' 'Online TA' 'Offline TA/TO' 'Complementary' 'Groups'
 'Undefined' 'Aviation']
distribution_channel
['Direct' 'Corporate' 'TA/TO' 'Undefined' 'GDS']
reserved_room_type
['C' 'A' 'D' 'E' 'G' 'F' 'H' 'L' 'P' 'B']
assigned_room_type
['C' 'A' 'D' 'E' 'G' 'F' 'I' 'B' 'H' 'P' 'L' 'K']
deposit_type
['No Deposit' 'Refundable' 'Non Refund']
customer_type
['Transient' 'Contract' 'Transient-Party' 'Group']
reservation_status
['Check-Out' 'Canceled' 'No-Show']
```

checking for missing values

In [17]:
```
df.isnull().sum() # this will return column names and how many missing values it has
```

Out[17]:
```
hotel                               0
is_canceled                         0
lead_time                           0
arrival_date_year                   0
arrival_date_month                  0
arrival_date_week_number            0
arrival_date_day_of_month           0
stays_in_weekend_nights             0
stays_in_week_nights                0
adults                              0
children                            4
babies                              0
meal                                0
country                           488
market_segment                      0
distribution_channel                0
is_repeated_guest                   0
previous_cancellations              0
previous_bookings_not_canceled      0
reserved_room_type                  0
assigned_room_type                  0
booking_changes                     0
deposit_type                        0
agent                           16340
company                        112593
days_in_waiting_list                0
customer_type                       0
adr                                 0
required_car_parking_spaces         0
total_of_special_requests           0
reservation_status                  0
reservation_status_date             0
dtype: int64
```

In [18]:
```
#will be difficult to handle columns 'agent' and 'company', too many missing values
```

In [19]:
```
df.drop(['company', 'agent'], axis = 1, inplace= True)
df.dropna(inplace = True)
```

In [20]:
```
df.isnull().sum()
```

Out[20]:
```
hotel                           0
is_canceled                     0
lead_time                       0
arrival_date_year               0
arrival_date_month              0
arrival_date_week_number        0
arrival_date_day_of_month       0
stays_in_weekend_nights         0
stays_in_week_nights            0
adults                          0
children                        0
babies                          0
meal                            0
country                         0
market_segment                  0
distribution_channel            0
is_repeated_guest               0
previous_cancellations          0
previous_bookings_not_canceled  0
reserved_room_type              0
assigned_room_type              0
booking_changes                 0
deposit_type                    0
days_in_waiting_list            0
customer_type                   0
adr                             0
required_car_parking_spaces     0
total_of_special_requests       0
reservation_status              0
reservation_status_date         0
dtype: int64
```

summary stats of the numerical columns:

In [21]:
```
df.describe()
```

Out[21]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	s
count	118898.000000	118898.000000	118898.000000	118898.000000	118898.000000	
mean	0.371352	104.311435	2016.157656	27.166555	15.800880	
min	0.000000	0.000000	2015.000000	1.000000	1.000000	
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	
75%	1.000000	161.000000	2017.000000	38.000000	23.000000	
max	1.000000	737.000000	2017.000000	53.000000	31.000000	
std	0.483168	106.903309	0.707459	13.589971	8.780324	

In [22]:
```
df = df[df['adr'] <300]
```

Analysing and Visualising Data

we want the see the amount of reservations that have been cancelled and the amount that hasnt been cancelled, lets look at count, precentage

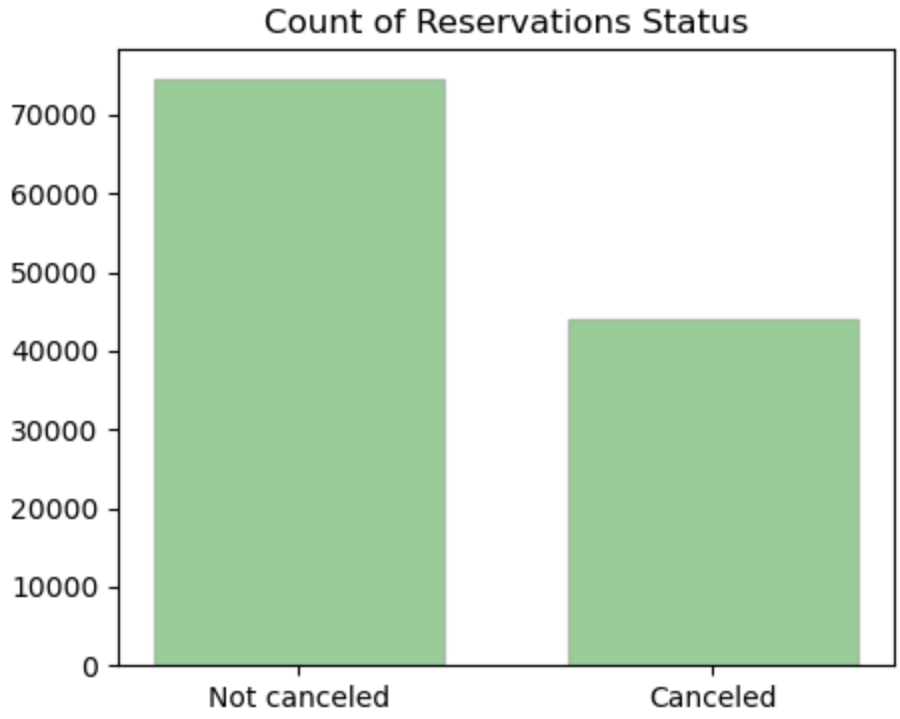
In [23]:
```
cancellation_perc= df['is_canceled'].value_counts(normalize= True) #this will return a %
ation_perc)
```

```
is_canceled
0    0.628494
1    0.371506
Name: proportion, dtype: float64
```

In [24]:
```
#we see that around 37% of reservations are cancelled,
#lets visualise this finding
```

In [25]:
```
print(cancellation_perc)
plt.figure(figsize = (5,4))
plt.title('Count of Reservations Status')
plt.bar(['Not canceled', 'Canceled'], df['is_canceled'].value_counts(),color='green', ed
plt.show()
```

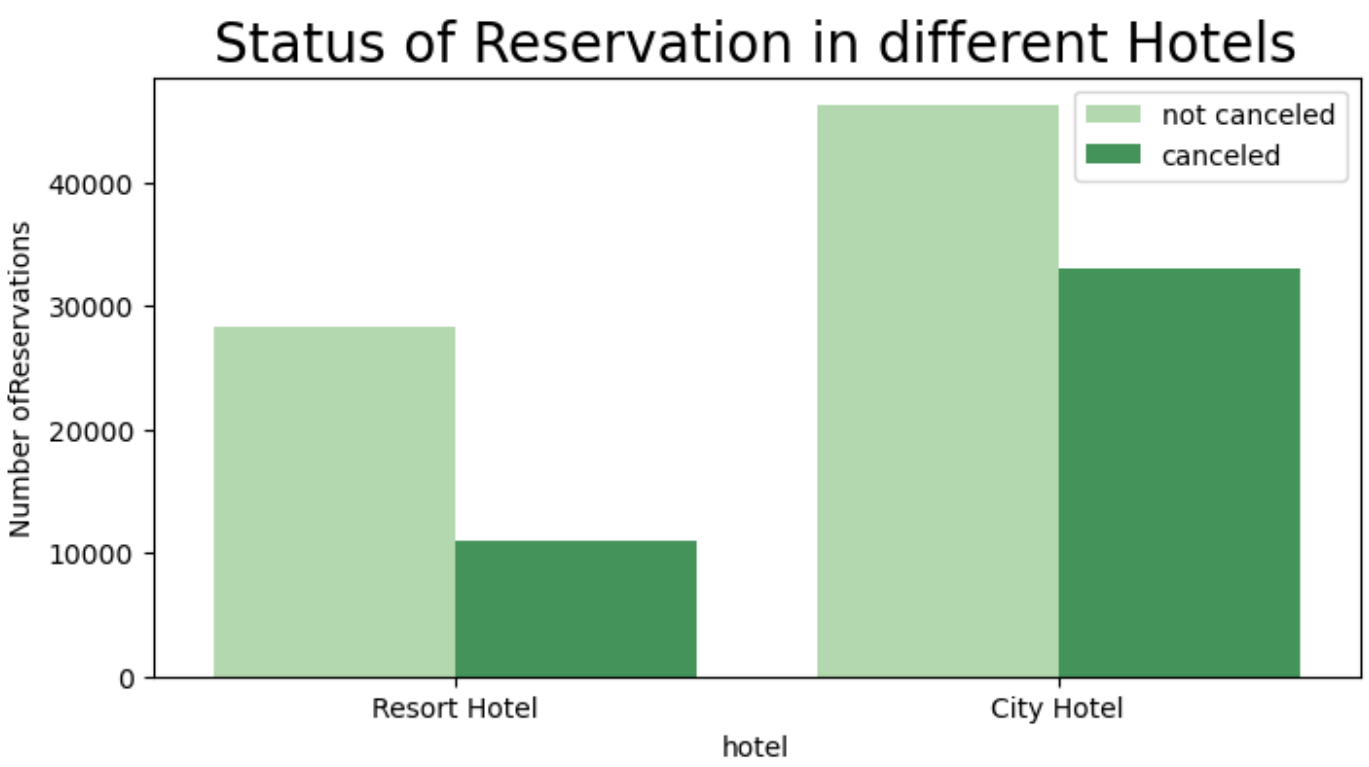
```
is_canceled
0    0.628494
1    0.371506
Name: proportion, dtype: float64
```



Can see that cancelled reservations amount for more than half of the non canceled reservations

Which hotel has the higher cancellation rate:

In [26]:
```
plt.figure(figsize=(8,4))
ax1= sns.countplot(x = 'hotel', hue = 'is_canceled', data = df, palette = 'Greens')
legend_labels = ax1.get_legend_handles_labels()
ax1.legend(labels=legend_labels[1], bbox_to_anchor=(1, 1))
plt.title('Status of Reservation in different Hotels', size = 20)
plt.xlabel('hotel')
plt.ylabel('Number ofReservations')
plt.legend(['not canceled', 'canceled'])
plt.show()
```



In [27]:
```
resort_hotel = df[df['hotel'] == 'Resort Hotel']
resort_hotel['is_canceled'].value_counts(normalize = True)
```

Out[27]:
```
is_canceled
0    0.720496
1    0.279504
Name: proportion, dtype: float64
```

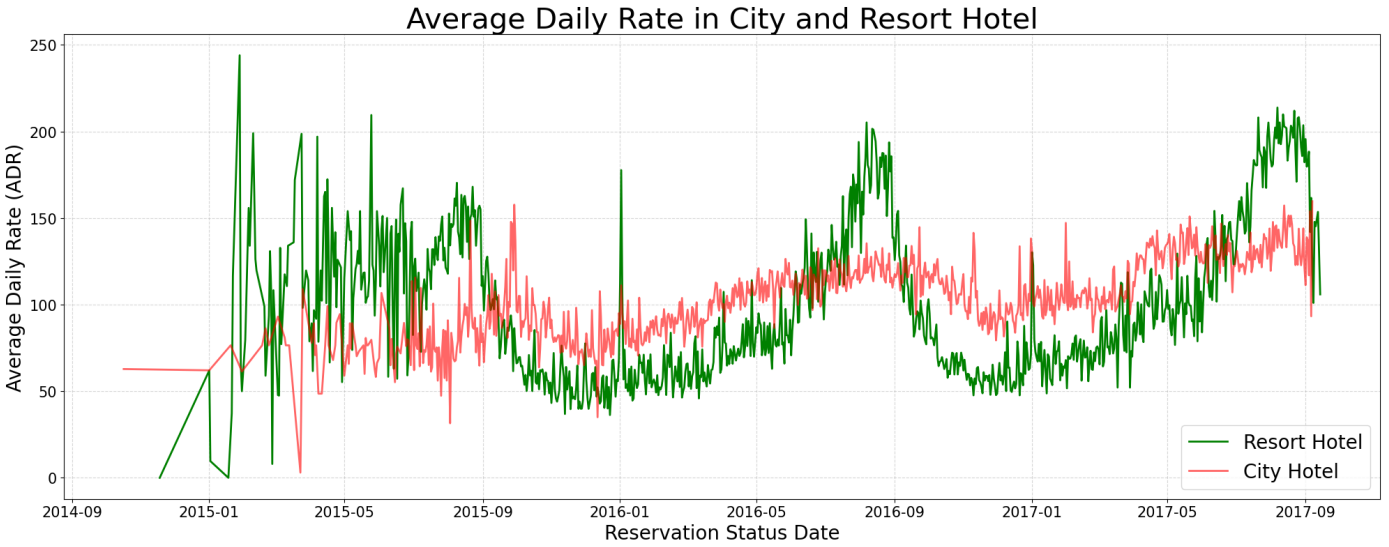
In [28]:
```
city_hotel = df[df['hotel'] == 'City Hotel']
city_hotel['is_canceled'].value_counts(normalize = True)
```

Out[28]:
```
is_canceled
0    0.582801
1    0.417199
Name: proportion, dtype: float64
```

In [48]:
```
resort_hotel = resort_hotel.groupby('reservation_status_date')[['adr']].mean()
city_hotel = city_hotel.groupby('reservation_status_date')[['adr']].mean()
```

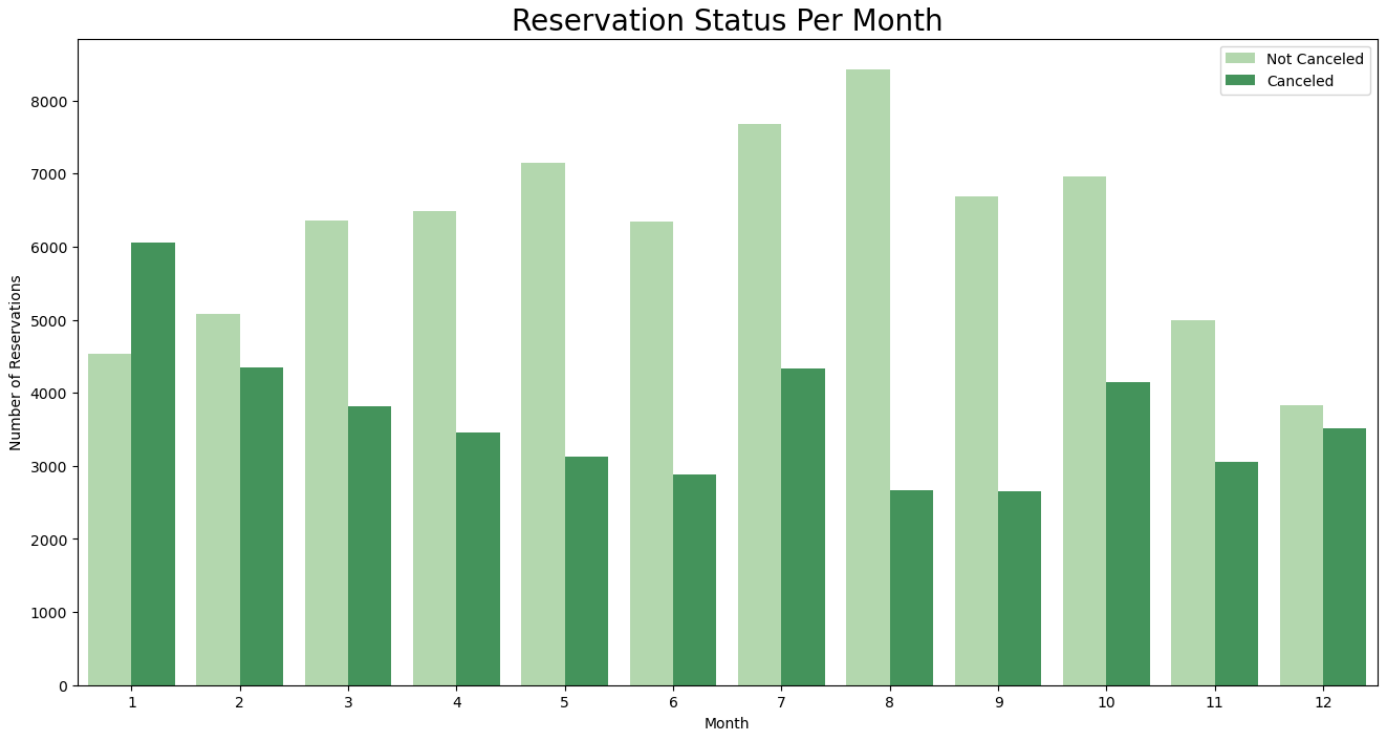
In [30]:
```
average_daily_rate = df.groupby(['reservation_status_date', 'hotel'])['adr'].mean().rese

resort_hotel = df[df['hotel'] == 'Resort Hotel']
city_hotel = df[df['hotel'] == 'City Hotel']


# Sort the data for each hotel by 'reservation_status_date'
resort_hotel = average_daily_rate[average_daily_rate['hotel'] == 'Resort Hotel']
city_hotel = average_daily_rate[average_daily_rate['hotel'] == 'City Hotel']

# Plot the data
plt.figure(figsize=(20, 8))
plt.title('Average Daily Rate in City and Resort Hotel', fontsize=30)
plt.plot(resort_hotel['reservation_status_date'], resort_hotel['adr'], label='Resort Hot
plt.plot(city_hotel['reservation_status_date'], city_hotel['adr'], label='City Hotel', c
plt.xlabel('Reservation Status Date', fontsize=20)
plt.ylabel('Average Daily Rate (ADR)', fontsize=20)
plt.legend(fontsize=20)
plt.grid(True, linestyle='--', alpha=0.5)
plt.xticks(fontsize=15)
```

```
plt.yticks(fontsize=15)
plt.tight_layout()
plt.show()
```



In [31]:
```
# the price of city hotel is less in comparison to resort hotelfor some days
#which month has the highest reservations and cancellations
```

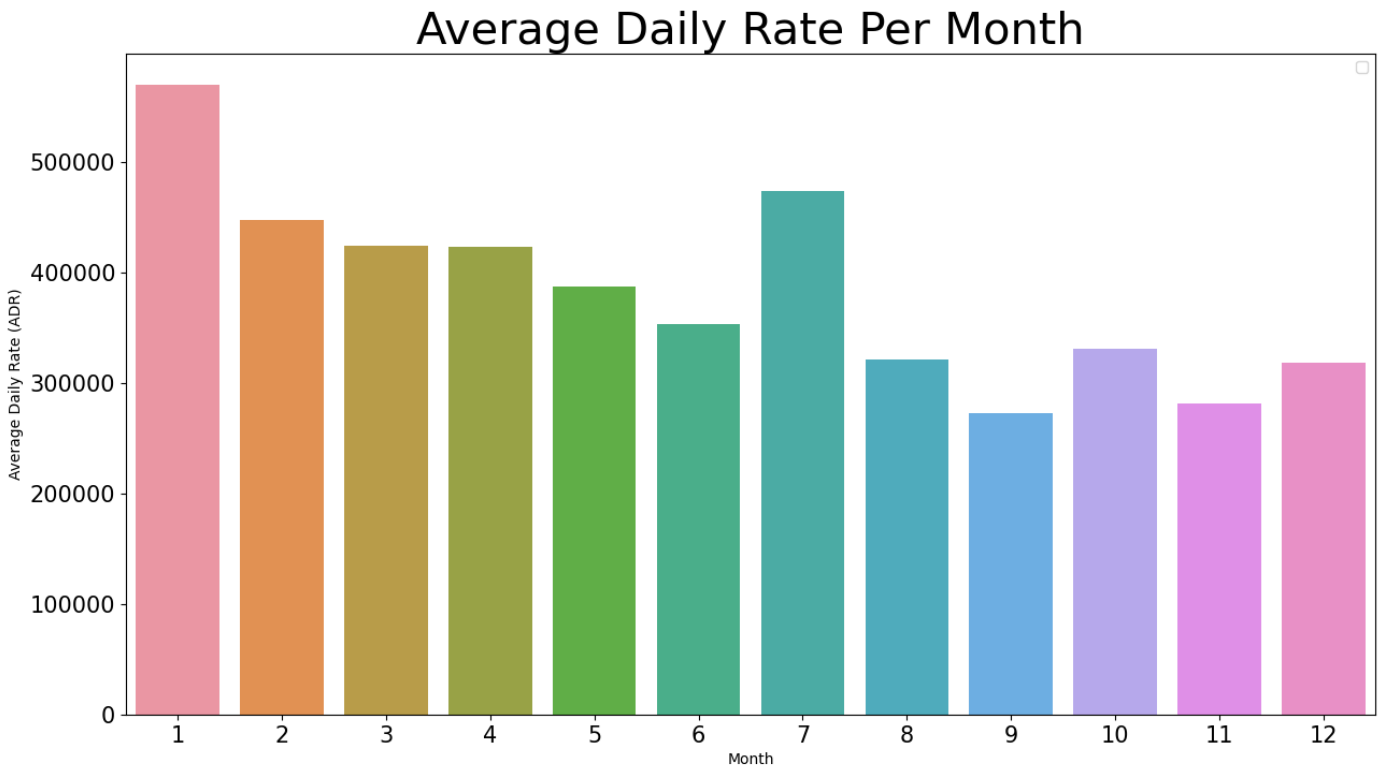
In [32]:
```
df['month'] = df['reservation_status_date'].dt.month

plt.figure(figsize=(16, 8))
ax1 = sns.countplot(x='month', hue='is_canceled', data=df, palette='Greens')
legend_labels, _ = ax1.get_legend_handles_labels()
ax1.legend(legend_labels, bbox_to_anchor=(1, 1))
plt.title('Reservation Status Per Month', size=20)
plt.xlabel('Month')
plt.ylabel('Number of Reservations')
plt.legend(['Not Canceled', 'Canceled'])
plt.show()
```



highest cancellation in january, lowestest cancellation when reservation is highest eg for august, were prices for the hotels lower? or was the price higher in january hence the higher cancellation?

In [49]:
```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap


plt.figure(figsize = (15,8))
plt.title( 'Average Daily Rate Per Month', fontsize = 30)
sns.barplot(x='month', y='adr', data=df[df['is_canceled'] == 1].groupby('month')[['adr']
plt.xlabel('Month')
plt.ylabel('Average Daily Rate (ADR)')
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.legend()
plt.show()
```

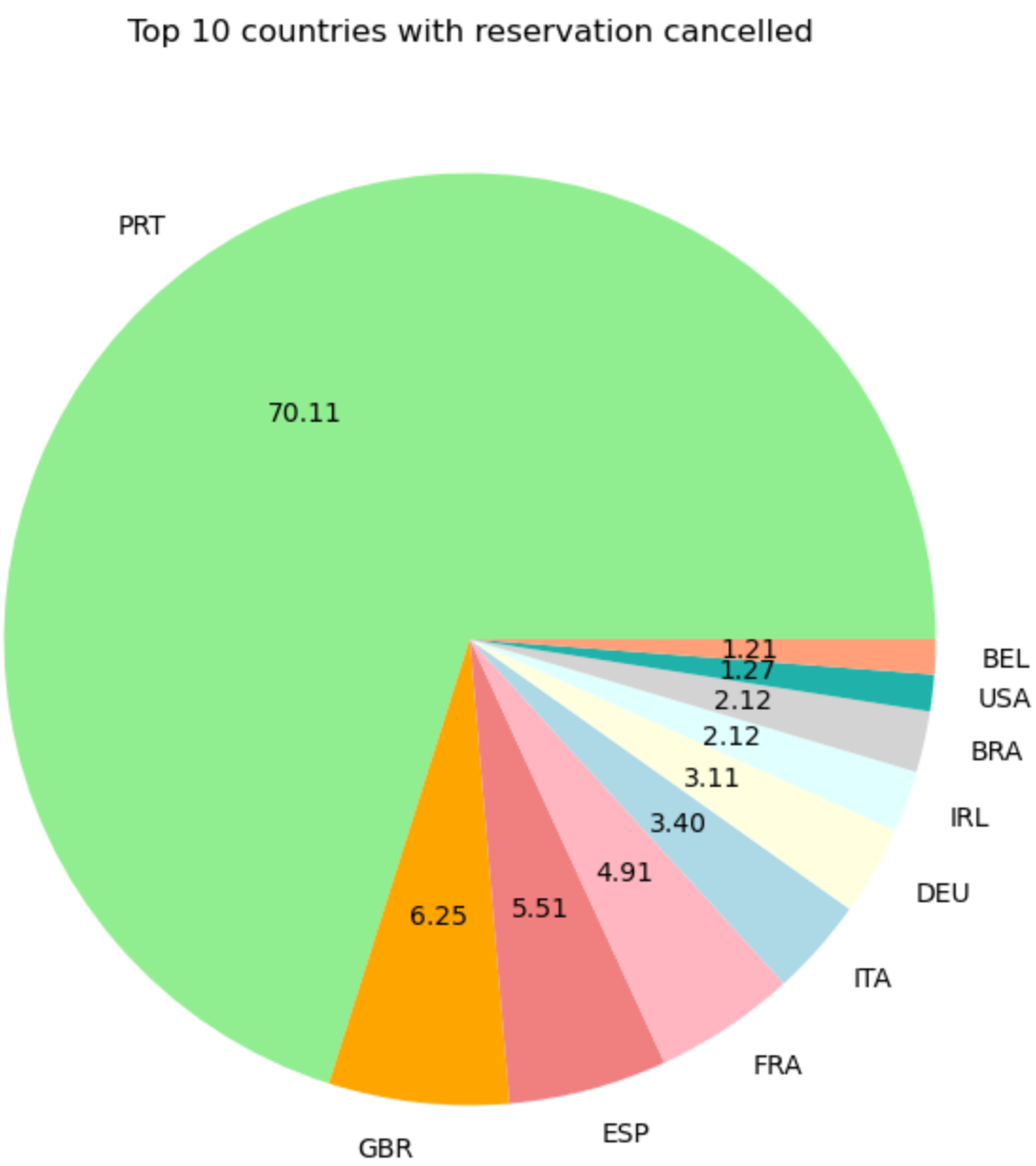
```
No artists with labels found to put in legend.  Note that artists whose label start with
an underscore are ignored when legend() is called with no argument.
```



proves the hypothesis that if the prices of the hotels are high then cancellations wil be higher cancellation rates based on country, top 10

In [34]:
```
cancelled_data =df[df['is_canceled'] ==1]
top_10_countries = cancelled_data['country'].value_counts()[:10]

# Define custom color palette
custom_colors = ['lightgreen','orange', 'lightcoral', 'lightpink', 'lightblue',
                 'lightyellow', 'lightcyan', 'lightgray', 'lightseagreen', 'lightsalmon'

plt.figure(figsize = (8,8))
plt.title('Top 10 countries with reservation cancelled')
plt.pie(top_10_countries, autopct = '%.2f', labels = top_10_countries.index, colors= cus
```

Out[34]:
```
([<matplotlib.patches.Wedge at 0x19fbd916cd0>,
  <matplotlib.patches.Wedge at 0x19fbe9114d0>,
  <matplotlib.patches.Wedge at 0x19fbd980e90>,
  <matplotlib.patches.Wedge at 0x19fbd983e10>,
  <matplotlib.patches.Wedge at 0x19fbd9786d0>,
  <matplotlib.patches.Wedge at 0x19fbd8d91d0>,
  <matplotlib.patches.Wedge at 0x19fbd90b910>,
  <matplotlib.patches.Wedge at 0x19fbd908210>,
  <matplotlib.patches.Wedge at 0x19fbd979010>,
  <matplotlib.patches.Wedge at 0x19fbd8d89d0>],
 [Text(-0.6495775459974918, 0.8877212466398878, 'PRT'),
  Text(-0.12200625266771079, -1.0932129135305633, 'GBR'),
  Text(0.280830681898992, -1.0635478964786433, 'ESP'),
  Text(0.6079111906640359, -0.9167573202682564, 'FRA'),
  Text(0.8239506782837557, -0.7287697028250689, 'ITA'),
  Text(0.9547297685327095, -0.5463433618865329, 'DEU'),
  Text(1.031244973338859, -0.3827973418969058, 'IRL'),
  Text(1.0729548740336765, -0.2424207876552208, 'BRA'),
  Text(1.092638943096332, -0.12704385081274458, 'USA'),
  Text(1.0992093885593792, -0.041697962814937276, 'BEL')],
 [Text(-0.35431502508954094, 0.4842115890763024, '70.11'),
  Text(-0.0665488650914786, -0.5962979528348527, '6.25'),
  Text(0.15318037194490472, -0.5801170344428963, '5.51'),
  Text(0.33158792218038313, -0.5000494474190489, '4.91'),
  Text(0.44942764270023033, -0.3975107469954921, '3.40'),
  Text(0.5207616919269324, -0.298005470119927, '3.11'),
  Text(0.562497258184832, -0.20879855012558496, '2.12'),
  Text(0.585248113109278, -0.13222952053921133, '2.12'),
  Text(0.5959848780525446, -0.06929664589786066, '1.27'),
  Text(0.599568757396025, -0.022744343353602148, '1.21')])
```



highest cancellation are in portugal, they need to implement better marketing strategies, promotional discounts - need to decrease cancellations

clients, where are they coming from, hypothesis stated they're likely coming from offline ta to make their reservations

In [35]:
```
df['market_segment'].value_counts()
```

Out[35]:
```
market_segment
Online TA        56233
Offline TA/TO    24152
Groups           19780
Direct           12364
Corporate         5109
Complementary      734
Aviation           237
Name: count, dtype: int64
```

In [36]:
```
df['market_segment'].value_counts(normalize = True)
```

Out[36]:
```
market_segment
Online TA        0.474104
Offline TA/TO    0.203627
Groups           0.166766
Direct           0.104242
Corporate        0.043074
Complementary    0.006188
Aviation         0.001998
Name: proportion, dtype: float64
```

most customers come from the online TA now lets see what the number of cancelations are coming from Online TA

In [37]:
```
cancelled_data['market_segment'].value_counts(normalize = True)
```

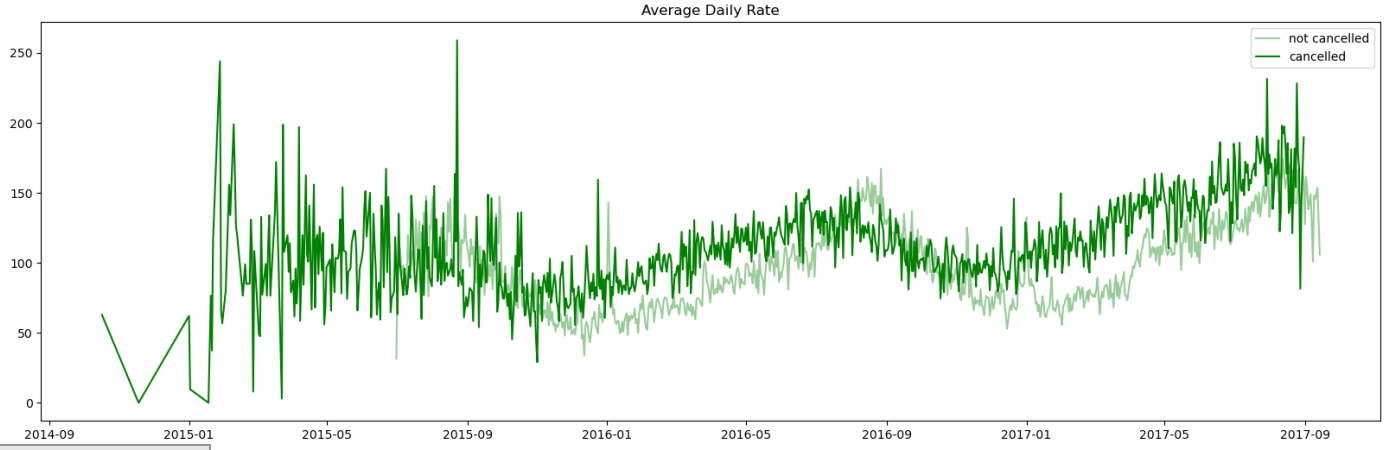
Out[37]:
```
market_segment
Online TA        0.469090
Groups           0.274510
Offline TA/TO    0.187840
Direct           0.043164
Corporate        0.022172
Complementary    0.002042
Aviation         0.001180
Name: proportion, dtype: float64
```

around 47% of cancellation is coming from the online TA possible reasons for cancellations online could be lack of appeal of marketing of the hotel rooms(photos) leading to customers cancelling reservations

lets see if the price (adr) of the cancelled are higher then the non cancelled

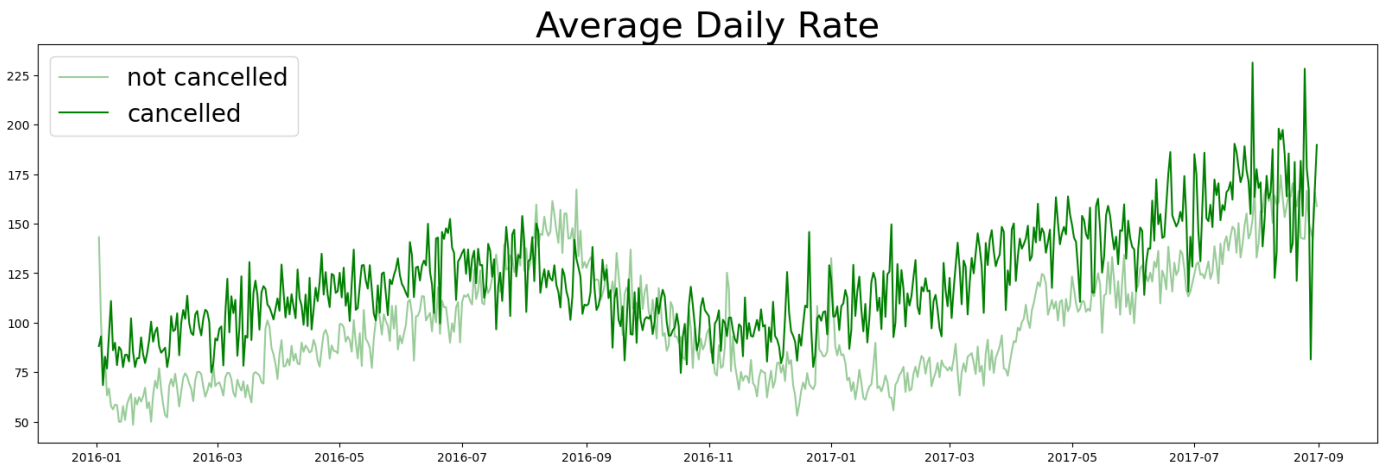
In [ ]:
```
```

In [42]:
```
cancelled_df_adr = cancelled_data.groupby('reservation_status_date')[['adr']].mean()
cancelled_df_adr.reset_index(inplace=True)
cancelled_df_adr.sort_values('reservation_status_date', inplace=True)

not_cancelled_data = df[df['is_canceled'] == 0]
not_cancelled_df_adr = not_cancelled_data.groupby('reservation_status_date')[['adr']].me
not_cancelled_df_adr.reset_index(inplace=True)
not_cancelled_df_adr.sort_values('reservation_status_date', inplace=True)  # Corrected s

plt.figure(figsize=(20, 6))
plt.title('Average Daily Rate')
plt.plot(not_cancelled_df_adr['reservation_status_date'], not_cancelled_df_adr['adr'], l
plt.plot(cancelled_df_adr['reservation_status_date'], cancelled_df_adr['adr'], label='ca
plt.legend()
plt.show()
```



In [43]:
```
cancelled_df_adr = cancelled_df_adr[(cancelled_df_adr['reservation_status_date'] > '2016
not_cancelled_df_adr = not_cancelled_df_adr[(not_cancelled_df_adr['reservation_status_da
```

In [46]:
```
plt.figure(figsize=(20, 6))
plt.title('Average Daily Rate', fontsize=30)
plt.plot(not_cancelled_df_adr['reservation_status_date'], not_cancelled_df_adr['adr'], l
plt.plot(cancelled_df_adr['reservation_status_date'], cancelled_df_adr['adr'], label='ca
plt.legend(fontsize=20)
plt.show()
```



price does have an influence on reservation cancellation rates

In [ ]:
```
```