Indian Food - Exploratory Data Analysis

Data Preparation and Cleaning:

In [1]:
```
# ----------------------------
#Importing library: To read dataset.
import pandas as pd
```

In [2]:
```
# ----------------------------
dset = pd.read_csv("C:\\Users\\HOME\\Python Sessions Jupyter\\Python - Data Analysis Project\\Indian_food.csv")
```

In [3]:
```
#To show dataset.
dset
```

Out[3]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
0	Balu shahi	Maida flour, yogurt, oil, sugar	vegetarian	45	25	sweet	dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	vegetarian	80	30	sweet	dessert	Rajasthan	West
2	Gajar ka halwa	Carrots, milk, sugar, ghee, cashews, raisins	vegetarian	15	60	sweet	dessert	Punjab	North
3	Ghevar	Flour, ghee, kewra, milk, clarified butter, su...	vegetarian	15	30	sweet	dessert	Rajasthan	West
4	Gulab jamun	Milk powder, plain flour, baking powder, ghee,...	vegetarian	15	40	sweet	dessert	West Bengal	East
...	...	...	...	...	...	...	...	...	...
250	Til Pitha	Glutinous rice, black sesame seeds, gur	vegetarian	5	30	sweet	dessert	Assam	North East

In [4]:
```
#To show the no. of rows and columns.
dset.shape
```

Out[4]:
```
(255, 9)
```

In [5]:
```
#To show no. of total values(elements) in the dataset.
dset.size
```

Out[5]:
```
2295
```

In [6]:
```
#To show indexes, columns, data-types of each column, memory at once.
dset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 255 entries, 0 to 254
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   name            255 non-null    object
 1   ingredients     255 non-null    object
 2   diet            255 non-null    object
 3   prep_time       255 non-null    int64 
 4   cook_time       255 non-null    int64 
 5   flavor_profile  255 non-null    object
 6   course          255 non-null    object
 7   state           255 non-null    object
 8   region          254 non-null    object
dtypes: int64(2), object(7)
memory usage: 18.1+ KB
```

In [7]:
```
# ----------------------------
#Entries with preparation time and cooking time greater than 0
new_dset=dset[dset.prep_time>0]
new_dset=new_dset[new_dset.cook_time>0]
```

In [8]:
```
new_dset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 225 entries, 0 to 253
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   name            225 non-null    object
 1   ingredients     225 non-null    object
 2   diet            225 non-null    object
 3   prep_time       225 non-null    int64 
 4   cook_time       225 non-null    int64 
 5   flavor_profile  225 non-null    object
 6   course          225 non-null    object
 7   state           225 non-null    object
 8   region          224 non-null    object
dtypes: int64(2), object(7)
memory usage: 17.6+ KB
```

In [9]:
```
#Find columns that contain at least one NaN:
print(new_dset.isnull().any())
```

```
name             False
ingredients      False
diet             False
prep_time        False
cook_time        False
flavor_profile   False
course           False
state            False
region            True
dtype: bool
```

In [10]:
```
#Find columns that contain at least one NaN:
# *Print that column name and values.
print(new_dset.loc[:, new_dset.isnull().any()])
```

```
         region
0          East
1          West
2         North
3          West
4          East
..          ...
247        East
249        East
250  North East
251        West
253     Central
```

In [11]:
```
# -----------------------------
#To show the count of NULL(NaN or Blank value) values in each column.
new_dset.isnull().sum()
```

Out[11]:
```
name             0
ingredients      0
diet             0
prep_time        0
cook_time        0
flavor_profile   0
course           0
state            0
region           1
dtype: int64
```

In [12]:
```
# -----------------------------
#Find rows that contain at least one NULL(NaN or Blank value):
new_dset[new_dset.isnull().any(axis=1)]
```

Out[12]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
110	Panjeeri	Whole wheat flour, musk melon seeds, poppy see...	vegetarian	10	25	sweet	dessert	Uttar Pradesh	NaN

In [13]:
```
# -----------------------------
#This method replaces the NULL values or NaN(blank) values with a specified value.
new_dset['region']=new_dset['region'].fillna('NA')
```

In [14]:
```
# -----------------------------
#To show all records of a perticular string in any column.
new_dset[new_dset['region'].str.contains('NA')]
```

Out[14]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
110	Panjeeri	Whole wheat flour, musk melon seeds, poppy see...	vegetarian	10	25	sweet	dessert	Uttar Pradesh	NA

In [15]:
```
new_dset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 225 entries, 0 to 253
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   name            225 non-null    object
 1   ingredients     225 non-null    object
 2   diet            225 non-null    object
 3   prep_time       225 non-null    int64
 4   cook_time       225 non-null    int64
 5   flavor_profile  225 non-null    object
 6   course          225 non-null    object
 7   state           225 non-null    object
 8   region          225 non-null    object
dtypes: int64(2), object(7)
memory usage: 17.6+ KB
```

In [16]:
```
#To check unique values of perticular column.
new_dset['diet'].unique()
```

Out[16]:
```
array(['vegetarian', 'non vegetarian'], dtype=object)
```

In [17]:
```
# ------------------------------
#Replace multiple values in a single column with different values.
new_dset['diet'].replace({'vegetarian':'Vegetarian', 'non vegetarian':'Non vegetarian'}, inplace=True)
```

In [18]:
```
new_dset.head(2)
```

Out[18]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
0	Balu shahi	Maida flour, yogurt, oil, sugar	Vegetarian	45	25	sweet	dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	Vegetarian	80	30	sweet	dessert	Rajasthan	West

In [19]:
```
new_dset['flavor_profile'].unique()
```

Out[19]:
```
array(['sweet', 'spicy', 'bitter', '-1', 'sour'], dtype=object)
```

In [20]:
```
# ------------------------------
#Rename column values.
new_dset['flavor_profile'].replace({'sweet':'Sweet','spicy':'Spicy','bitter':'Bitter','-1':'NA','sour':'Sour'}, inplace=True)
```

In [21]:
```
new_dset.head(2)
```

Out[21]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
0	Balu shahi	Maida flour, yogurt, oil, sugar	Vegetarian	45	25	Sweet	dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	Vegetarian	80	30	Sweet	dessert	Rajasthan	West

In [22]:
```
new_dset['course'].unique()
```

Out[22]:
```
array(['dessert', 'main course', 'starter', 'snack'], dtype=object)
```

In [23]:
```
# ------------------------------
new_dset['course'].replace({'dessert':'Dessert','main course':'Main course','starter':'Starter','snack':'Snack'}, inplace=True)
```

In [24]:
```
new_dset.head(2)
```

Out[24]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
0	Balu shahi	Maida flour, yogurt, oil, sugar	Vegetarian	45	25	Sweet	Dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	Vegetarian	80	30	Sweet	Dessert	Rajasthan	West

In [25]:
```
new_dset['state'].unique()
```

Out[25]:
```
array(['West Bengal', 'Rajasthan', 'Punjab', 'Uttar Pradesh', '-1',
       'Odisha', 'Maharashtra', 'Uttarakhand', 'Assam', 'Bihar',
       'Andhra Pradesh', 'Karnataka', 'Telangana', 'Kerala', 'Tamil Nadu',
       'Gujarat', 'Manipur', 'Nagaland', 'NCT of Delhi',
       'Jammu & Kashmir', 'Chhattisgarh', 'Haryana', 'Madhya Pradesh',
       'Goa'], dtype=object)
```

In [26]:
```
# ------------------------------
new_dset['state'].replace({'-1':'NA'}, inplace=True)
```

In [27]:
```
new_dset.head(2)
```

Out[27]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
0	Balu shahi	Maida flour, yogurt, oil, sugar	Vegetarian	45	25	Sweet	Dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	Vegetarian	80	30	Sweet	Dessert	Rajasthan	West

In [28]:
```
new_dset['region'].unique()
```

Out[28]:
```
array(['East', 'West', 'North', '-1', 'North East', 'South', 'Central',
       'NA'], dtype=object)
```

In [29]:
```
# ------------------------------
new_dset['region'].replace({'-1':'NA'}, inplace=True)
```

In [30]:
```
new_dset.head(10)
```

Out[30]:

	name	ingredients	diet	prep_time	cook_time	flavor_profile	course	state	region
0	Balu shahi	Maida flour, yogurt, oil, sugar	Vegetarian	45	25	Sweet	Dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	Vegetarian	80	30	Sweet	Dessert	Rajasthan	West
2	Gajar ka halwa	Carrots, milk, sugar, ghee, cashews, raisins	Vegetarian	15	60	Sweet	Dessert	Punjab	North
3	Ghevar	Flour, ghee, kewra, milk, clarified butter, su...	Vegetarian	15	30	Sweet	Dessert	Rajasthan	West
4	Gulab jamun	Milk powder, plain flour, baking powder, ghee,...	Vegetarian	15	40	Sweet	Dessert	West Bengal	East
5	Imarti	Sugar syrup, lentil flour	Vegetarian	10	50	Sweet	Dessert	West Bengal	East
6	Jalebi	Maida, corn flour, baking soda, vinegar, curd,...	Vegetarian	10	50	Sweet	Dessert	Uttar Pradesh	North

In [31]:
```
# ------------------------------
#To rename multiple columns.
new_dset.rename(columns={'name':'Name','ingredients':'Ingredients','diet':'Diet','prep_time':'Prep Time','cook_time':'Cook Time',
```

In [32]:
```
new_dset.head(2)
```

Out[32]:

	Name	Ingredients	Diet	Prep Time	Cook Time	Flavour Profile	Course	State	Region
0	Balu shahi	Maida flour, yogurt, oil, sugar	Vegetarian	45	25	Sweet	Dessert	West Bengal	East
1	Boondi	Gram flour, ghee, sugar	Vegetarian	80	30	Sweet	Dessert	Rajasthan	West

In [33]:
```
#To count specific values across multiple columns.
new_dset[new_dset == -1].count()
```

Out[33]:
```
Name              0
Ingredients       0
Diet              0
Prep Time         0
Cook Time         0
Flavour Profile   0
Course            0
State             0
Region            0
dtype: int64
```

In [34]:
```
new_dset[new_dset == 0].count()
```

Out[34]:
```
Name              0
Ingredients       0
Diet              0
Prep Time         0
Cook Time         0
Flavour Profile   0
Course            0
State             0
Region            0
dtype: int64
```

In [35]:
```
new_dset[new_dset == 'NA'].count()
```

Out[35]:
```
Name              0
Ingredients       0
Diet              0
Prep Time         0
Cook Time         0
Flavour Profile  24
Course            0
State            23
Region           13
dtype: int64
```

In [36]:
```
# ------------------------------
#To check row wise and detect the duplicate rows.
new_dset[new_dset.duplicated()]
```

Out[36]:

Name	Ingredients	Diet	Prep Time	Cook Time	Flavour Profile	Course	State	Region

In [37]:
```
new_dset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 225 entries, 0 to 253
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Name            225 non-null    object
 1   Ingredients     225 non-null    object
 2   Diet            225 non-null    object
 3   Prep Time       225 non-null    int64 
 4   Cook Time       225 non-null    int64 
 5   Flavour Profile 225 non-null    object
 6   Course          225 non-null    object
 7   State           225 non-null    object
 8   Region          225 non-null    object
dtypes: int64(2), object(7)
memory usage: 17.6+ KB
```

Data Visualization

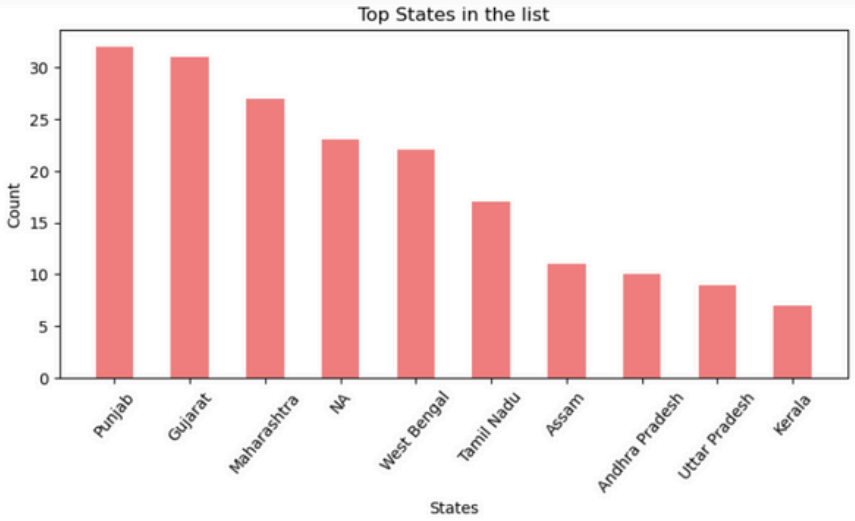
In [38]:
```
# ------------------------------
#Difference between matplotlib & matplotlib.pyplot:-
#*matplotlib: Importing all its libraries. *matplotlib.pyplot: Only imports pyplot's properties.
#Pyplot
import matplotlib.pyplot as plt
```

1)State wise number of Indian dishes?

In [39]:
```
x=new_dset['State'].value_counts().head(10)
x
```

Out[39]:
```
Punjab            32
Gujarat           31
Maharashtra       27
NA                23
West Bengal       22
Tamil Nadu        17
Assam             11
Andhra Pradesh    10
Uttar Pradesh      9
Kerala             7
Name: State, dtype: int64
```

In [40]:
```
plt.figure(figsize=(9,4))
plt.bar(x.index, x, width=0.5, color='lightcoral')
plt.xticks(rotation=50)
plt.title("Top States in the list")
plt.xlabel('States')
plt.ylabel('Count')
plt.show()
```

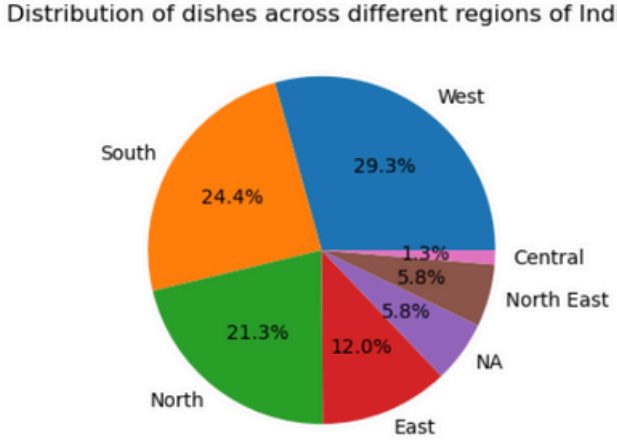


2) Distribution of dishes across different regions of India?

In [41]:
```
x_region=new_dset['Region'].value_counts()
x_region
```

Out[41]:
```
West          66
South         55
North         48
East          27
NA            13
North East    13
Central        3
Name: Region, dtype: int64
```

In [42]:
```
plt.figure(figsize=(6,4))
plt.pie(x_region, labels=x_region.index, autopct='%1.1f%%')
plt.title("Distribution of dishes across different regions of India")
plt.show()
```



3) Percentage of dishes available in various Flavour Profiles?

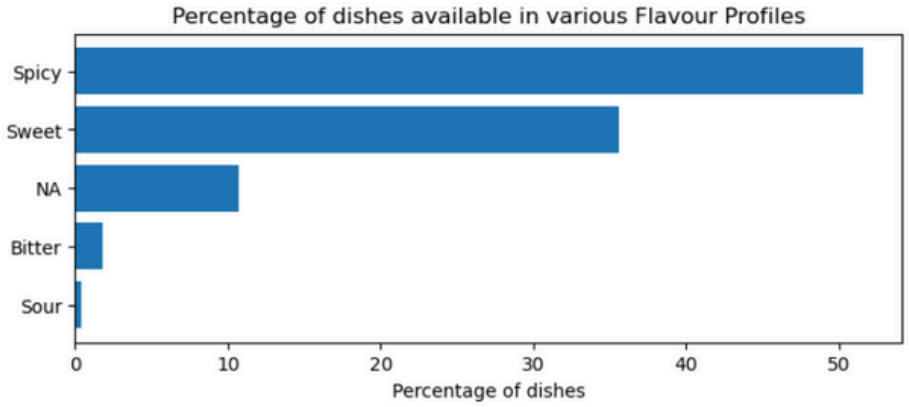
In [44]:
```
# new_dset['Flavour Profile'].value_counts().sort_values(ascending=True)
```

Out[44]:
```
Sour        1
Bitter      4
NA         24
Sweet      80
Spicy     116
Name: Flavour Profile, dtype: int64
```

In [45]:
```
# new_dset['Flavour Profile'].count()
```

Out[45]:
```
225
```

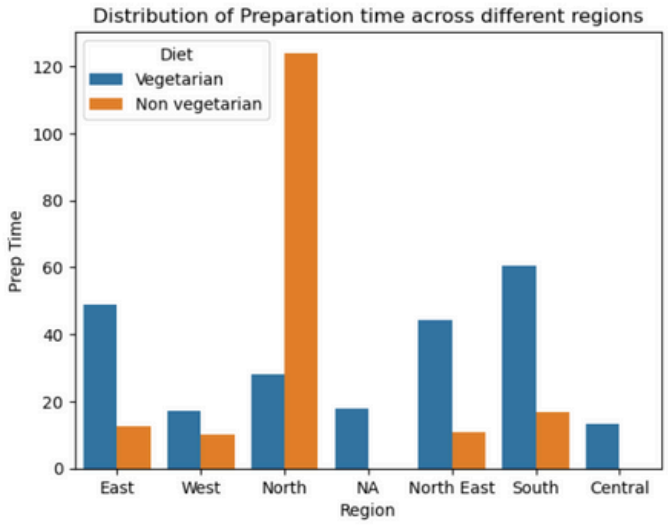
In [46]:
```
flavor = new_dset['Flavour Profile'].value_counts().sort_values(ascending=True) * 100/ new_dset['Flavour Profile'].count()
plt.figure(figsize=(8,3))
plt.barh(flavor.index, flavor)
plt.title("Percentage of dishes available in various Flavour Profiles")
plt.xlabel('Percentage of dishes')
plt.show()
```



4) Comparing Preparation time & Cooking time for Veg & Non-veg dishes?

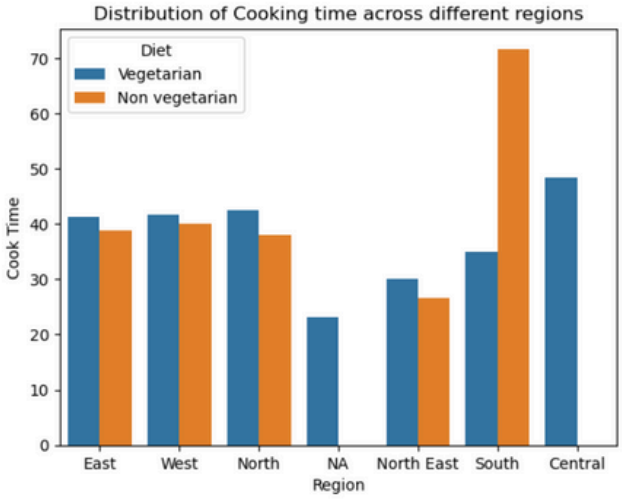
In [47]:
```
# ----------------------------
import seaborn as sns
plt.title("Distribution of Preparation time across different regions")
sns.barplot(x='Region',y='Prep Time',hue='Diet',data=new_dset,ci=None)
```

Out[47]:
```
<AxesSubplot:title={'center':'Distribution of Preparation time across different regions'}, xlabel='Region', ylabel='Prep Time'>
```



In [48]:
```
# Cooking Time
plt.title("Distribution of Cooking time across different regions")
sns.barplot(x='Region',y='Cook Time',hue='Diet',data=new_dset,ci=None)
```

Out[48]:
```
<AxesSubplot:title={'center':'Distribution of Cooking time across different regions'}, xlabel='Region', ylabel='Cook Time'>
```

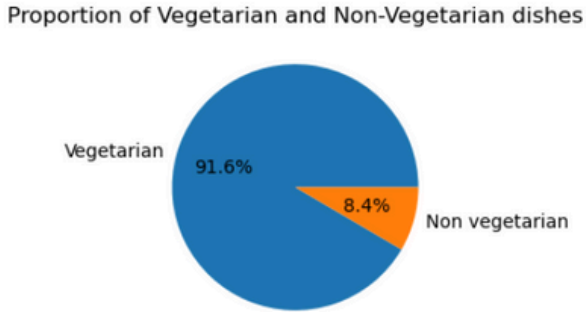


5) Proportion of Vegetarian and Non-Vegetarian dishes?

In [49]:
```
x_diet=new_dset['Diet'].value_counts()
x_diet
```

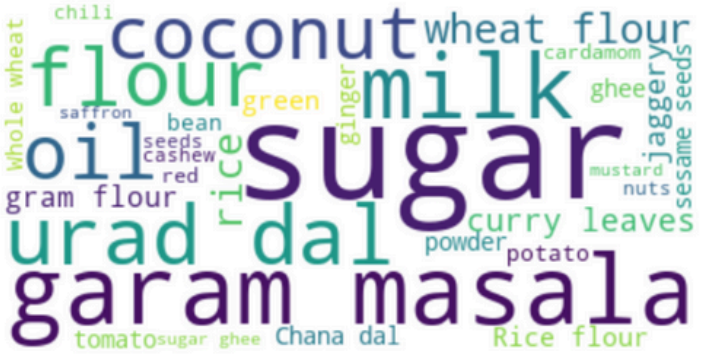
Out[49]:
```
Vegetarian        206
Non vegetarian     19
Name: Diet, dtype: int64
```

In [50]:
```
plt.figure(figsize=(6,3))
plt.pie(x_diet, labels=x_diet.index, autopct='%1.1f%%')
plt.title("Proportion of Vegetarian and Non-Vegetarian dishes")
#plt.legend(title="Regions:")
plt.show()
```



6) Ingredients used in Vegetarian food?

In [51]:
```
# ----------------------------
#Importing libraries for Word Cloud chart:
from wordcloud import WordCloud
veg_df=new_dset[new_dset['Diet']=='Vegetarian'].reset_index()
Ingredients=[]
for i in range(0,len(veg_df)):
    text=veg_df['Ingredients'][i].split(',')
    text=','.join(text)
    Ingredients.append(text)
    text=' '.join(Ingredients)
wordcloud=WordCloud(width=400, height=200, background_color='white', min_font_size=10).generate(text)
plt.figure(figsize=(8,8), facecolor=None)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

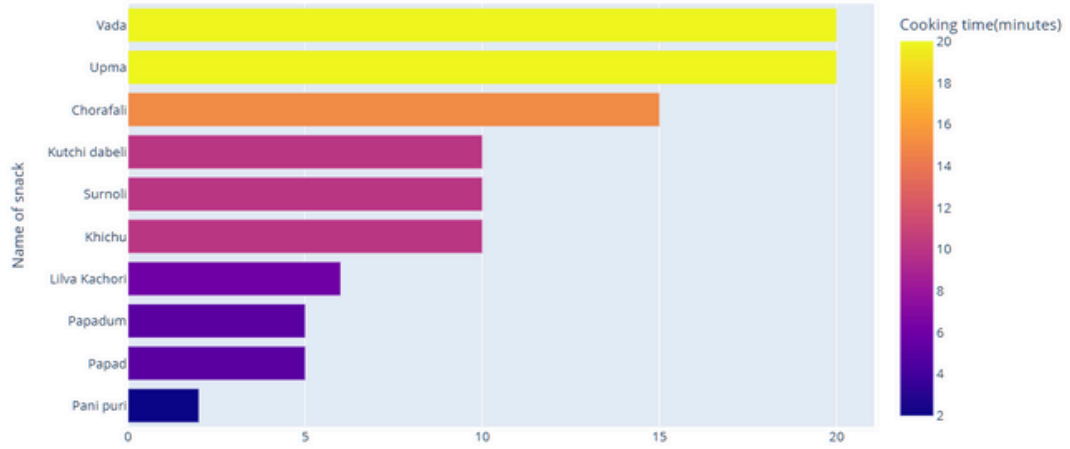


7) Ingredients used in Non-Vegetarian food?

In [52]:
```
from wordcloud import WordCloud
non_veg_df=new_dset[new_dset['Diet']=='Non vegetarian'].reset_index()
Ingredients=[]
for l in range(0,len(non_veg_df)):
    string=non_veg_df['Ingredients'][l].split(',')
    string=','.join(string)
    Ingredients.append(string)
    string=' '.join(Ingredients)
wordcloud=WordCloud(width=400, height=200, background_color='white', min_font_size=10).generate(string)
plt.figure(figsize=(8,8), facecolor=None)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

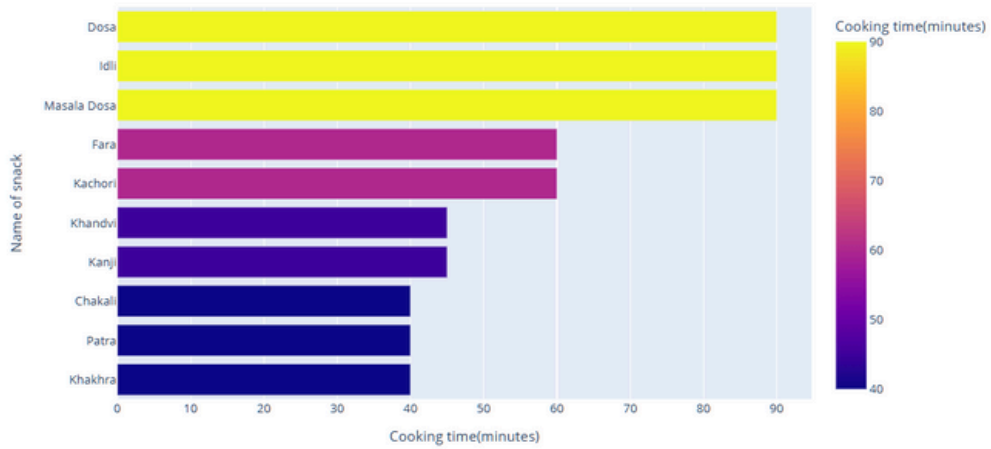
8) Top 10 Snacks with shortest cooking time?

In [53]:
```
import plotly.express as px
snack_df=new_dset[new_dset['Course']=='Snack']
short_sort_snack_df=snack_df.sort_values(['Cook Time'],ascending=True).iloc[:10,:]
fig=px.bar(short_sort_snack_df,y='Name',x='Cook Time',orientation='h',color='Cook Time',labels={'Name':'Name of snack','Cook Time
fig.show()
```



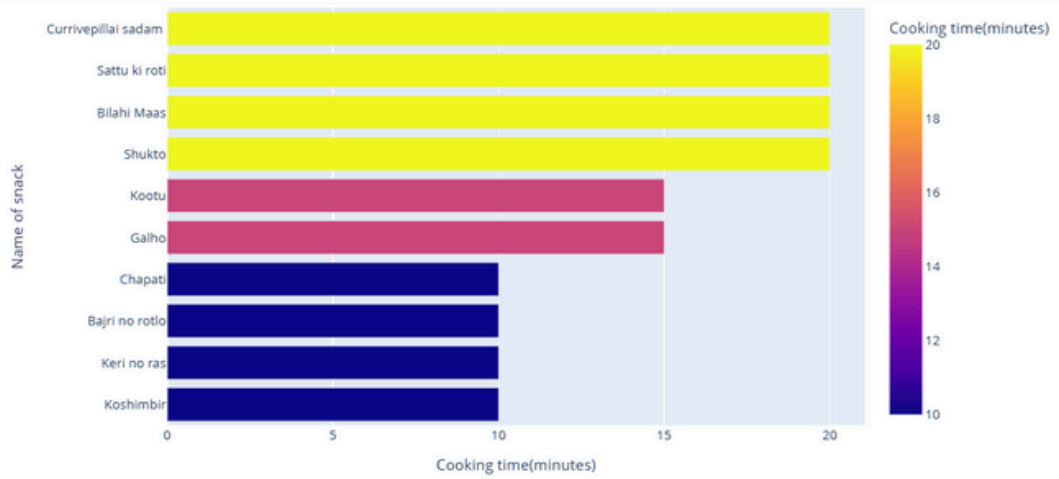
9) Top 10 Snacks with longest cooking time?

In [54]:
```
# snack_df=new_dset[new_dset['Course']=='Snack']
long_sort_snack_df=snack_df.sort_values(['Cook Time'],ascending=True).iloc[26:36,:]
fig=px.bar(long_sort_snack_df,y='Name',x='Cook Time',orientation='h',color='Cook Time',labels={'Name':'Name of snack','Cook Time'
fig.show()
```



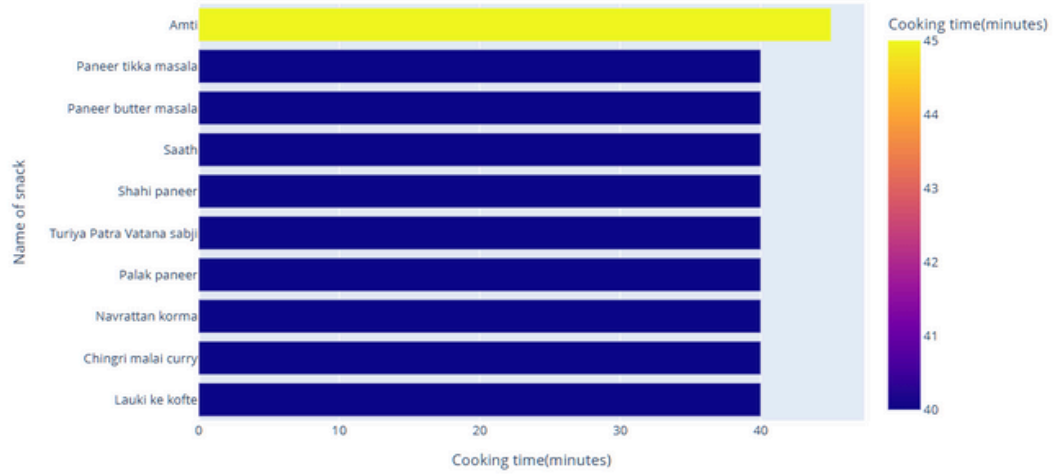
10) Top 10 Main Courses with shortest cooking time?

In [55]:
```
mc_df=new_dset[new_dset['Course']=='Main course']
small_mc_df=mc_df.sort_values(['Cook Time'],ascending=True).iloc[:10,:]
fig=px.bar(small_mc_df,y='Name',x='Cook Time',orientation='h',color='Cook Time',labels={'Name':'Name of snack','Cook Time':'Cooki
fig.show()
```



11) Top 10 Main Courses with longest cooking time?

In [56]:
```
long_mc_df=mc_df.sort_values(['Cook Time'],ascending=True).iloc[-30:-20,:]
fig=px.bar(long_mc_df,y='Name',x='Cook Time',orientation='h',color='Cook Time',labels={'Name':'Name of snack','Cook Time':'Cookir
fig.show()
```

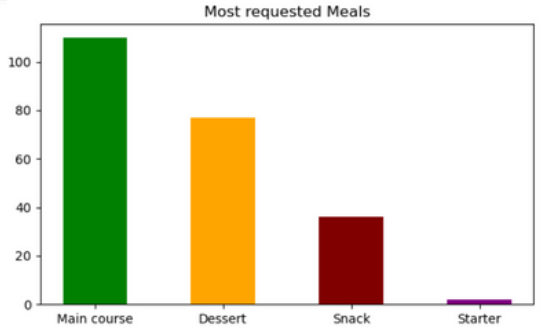


12) Most requested meals?

In [57]:
```
meal_dset=new_dset['Course'].value_counts()
meal_dset
```

Out[57]:
```
Main course    110
Dessert         77
Snack           36
Starter          2
Name: Course, dtype: int64
```

In [58]:
```
plt.figure(figsize=(7,4))
colors_list={'Purple','Orange','Maroon','Green'}
plt.bar(meal_dset.index, meal_dset, width=0.5, color=colors_list)
plt.title("Most requested Meals")
plt.ylabel('Count')
plt.show()
```



13) Correlation Heatmap?

In [59]:
```
print(new_dset.corr())
```

```
             Prep Time  Cook Time
Prep Time    1.00000    0.11078
Cook Time    0.11078    1.00000
```

In [60]:
```
dataheat=sns.heatmap(new_dset.corr(), cmap='YlGnBu',annot=True)
plt.show()
```