E-News Express

Objective:

Explore the dataset and extract insights from the data.

1. Determine whether the new landing page is more effective to gather new subscribers.
2. Perform uni-variate and multi-variate analyses;
3. Perform the statistical analysis and visual analysis;
4. Generate a set of insights and recommendations that will help the company in to expand its business by acquiring new subscribers.

Key Questions:

1. Do the users spend more time on the new landing page than the old landing page?
2. Is the conversion rate (the proportion of users who visit the landing page and get converted) for the new page greater than the conversion rate for the old page?
3. Does the converted status depend on the preferred language?

Data Dictionary:

The data is for 100 randomly selected users of a online news portal called E-news Express. It contains the following variables:

1. user_id - This represents the user ID of the person visiting the website.
2. group - This represents whether the user belongs to the first group (control) or the second group (treatment).
3. landing_page - This represents whether the landing page is new or old.
4. time_spent_on_the_page - This represents the time (in minutes) spent by the user on the landing page.
5. converted - This represents whether the user gets converted or not.
6. language_preferred - This represents the language chosen by the user to view the landing page.

Consider a significance level of 0.05 for all tests.

Importing Dataset

In [1]:
```
# Never print matching warnings (remove all warnings and errors)
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# Importing the necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
import seaborn as sns
from scipy import stats
```

In [3]:
```
# Reading the dataset
data = pd.read_csv('abtest.csv')
```

In [4]:
```
# Coping the data to another variable to avoid any changes to the original da
news = data.copy()
```

Understanding the structure of the dataset

In [5]:
```
# Visualizing 4 first rows of data
news.head(4)
```

Out[5]:

	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferred
0	546592	control	old	3.48	yes	Spanish
1	546468	treatment	new	7.13	yes	English
2	546462	treatment	new	4.40	no	Spanish
3	546567	control	old	3.02	no	French

In [6]:
```
# Visualizing 5 last rows of data
news.tail()
```

Out[6]:

	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferred
95	546446	treatment	new	5.15	no	Spanish
96	546544	control	old	6.52	yes	English
97	546472	treatment	new	7.07	yes	Spanish
98	546481	treatment	new	6.20	yes	Spanish
99	546483	treatment	new	5.86	yes	English

Observations

- User ID contains the ID of the person visiting the website.
- Group represents whether the user belongs to treatment(new landing page) or to control (old landing page)
- Landing_page represents whether the user belongs to new landing page (treatment group) or to old landing page (control group)
- Time spent on the page represents minutes that the user spent on the landing page
- Converted shows whether the user became subscriber or not on the landing page
- Language preferred shows which language the user prefers to.
- The variables: Group, Landing_page, converted and, language_preferred are categorical variable

In [7]:
```
# Shape of the dataset
news.shape
```

Out[7]:
```
(100, 6)
```

- The dataset has 100 rows and 6 columns

In [8]:
```
# Dataframe
news.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   user_id                 100 non-null    int64  
 1   group                   100 non-null    object 
 2   landing_page            100 non-null    object 
 3   time_spent_on_the_page  100 non-null    float64
 4   converted               100 non-null    object 
 5   language_preferred      100 non-null    object 
dtypes: float64(1), int64(1), object(4)
memory usage: 4.8+ KB
```

Observations:

- All columns have 100 observations non-null, which indicat that there are no missing values in it. *(treatment of missing values is not necessary)*.
- group, landing_page, converted and language_preferred should be categorical variables.

Data Preprocessing

In [9]:
```
# Converting object data type to categorical data type (It reduces the data s
news['group'] = news.group.astype('category')
news['landing_page'] = news.landing_page.astype('category')
news['converted'] = news.converted.astype('category')
news['language_preferred'] = news.language_preferred.astype('category')
```

In [10]:
```
news.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   user_id                 100 non-null    int64  
 1   group                   100 non-null    category
 2   landing_page            100 non-null    category
 3   time_spent_on_the_page  100 non-null    float64
 4   converted               100 non-null    category
 5   language_preferred      100 non-null    category
dtypes: category(4), float64(1), int64(1)
memory usage: 2.5 KB
```

In [11]:
```
# checking descriptive statistics
# include all means even the ones that is not numerical like categorical
news.describe(include='all')
```

Out[11]:

	user_id	group	landing_page	time_spent_on_the_page	converted	languag
count	100.000000	100	100	100.000000	100	
unique	NaN	2	2	NaN	2	
top	NaN	treatment	old	NaN	yes	
freq	NaN	50	50	NaN	55	

	user_id	group	landing_page	time_spent_on_the_page	converted	languag
mean	546517.000000	NaN	NaN	5.377800	NaN	
std	52.295779	NaN	NaN	2.378166	NaN	
min	546443.000000	NaN	NaN	0.190000	NaN	
25%	546467.750000	NaN	NaN	3.880000	NaN	
50%	546492.500000	NaN	NaN	5.415000	NaN	
75%	546567.250000	NaN	NaN	7.022500	NaN	
max	546592.000000	NaN	NaN	10.710000	NaN	

In [12]:
```
news.language_preferred.value_counts()
```

Out[12]:
```
Spanish    34
French     34
English    32
Name: language_preferred, dtype: int64
```

In [13]:
```
news.converted.value_counts()
```

Out[13]:
```
yes    55
no     45
Name: converted, dtype: int64
```

E-News Express users spent time on *Old* landing page

In [14]:
```
# Filter by landing page OLD
old_users = news[news['landing_page']=='old']

# Describe numerical variable
old_users_des = old_users.describe().T

# Droping row user id, because it is not 'necessary' for now
old_users_des.drop('user_id', inplace=True)

old_users_des
```

Out[14]:

	count	mean	std	min	25%	50%	75%	max
time_spent_on_the_page	50.0	4.5324	2.581975	0.19	2.72	4.38	6.4425	10.3

E-News Express users spent time on *New* landing page

In [15]:
```
# Filter by landing page NEW
new_users = news[news['landing_page']=='new']

# Describe numerical variable
new_users_des = new_users.describe().T

# Droping row user id, because it is not "necessary" for now
new_users_des.drop('user_id', inplace=True)

new_users_des
```

Out[15]:

	count	mean	std	min	25%	50%	75%	max
time_spent_on_the_page	50.0	6.2232	1.817031	1.65	5.175	6.105	7.16	10.71

Observation:

- Spent time mean on New landing page is greather than on Old page;
- Spread of data (std) on New langing page is less than on Old page, meaning that data is spread near by the mean.

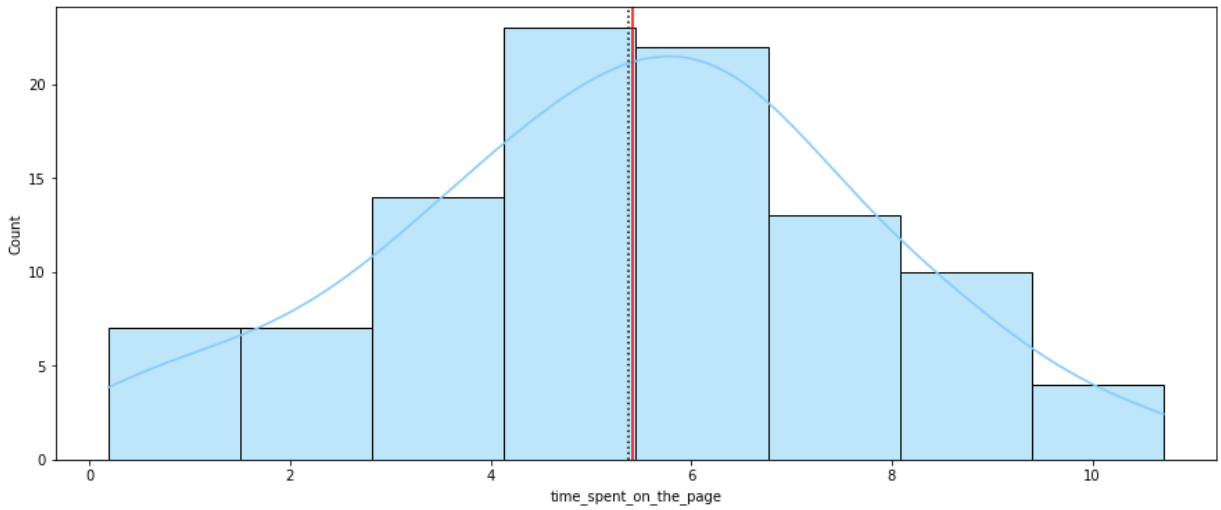
EDA

Uni-variate: Exploring the numerical variable

Users Time Spent - Distribution:

In [16]:
```
# Ploting a histplot for time spent on the page
plt.figure(figsize=(15,6))
ax_hist = sns.histplot(news['time_spent_on_the_page'],kde=True, color= 'light

# Add mean to the histogram
ax_hist.axvline(np.mean(news['time_spent_on_the_page']), color='black', lines
# Add median to the histogram
ax_hist.axvline(np.median(news['time_spent_on_the_page']), color='red', lines
```

Out[16]:
```
<matplotlib.lines.Line2D at 0x2719a262af0>
```



Observations:

- The histogram shows that data is approximately normally distributed.
- Mean and median it is pretty close to each other.

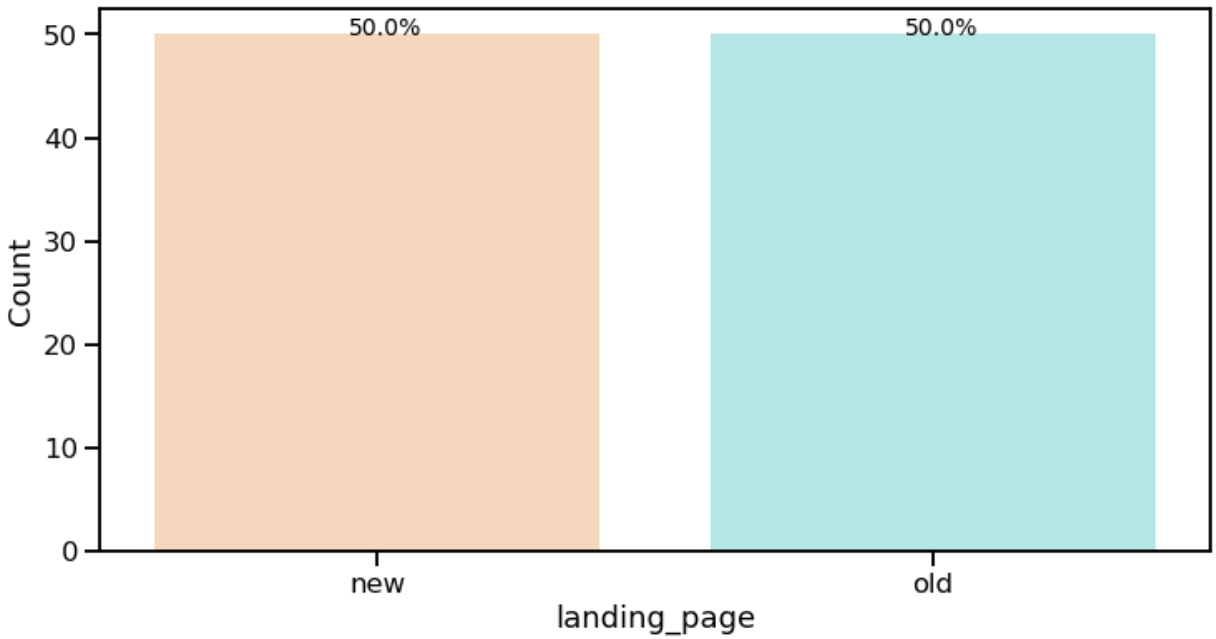
Uni-variate: Exploring the categorical variables

In [17]:
```
# Function to create barplots that indicate percentage for each category.

def bar_perc(dataframe, xlabel_df, colors, ylabel_df = 'Count'):
    """
    This function takes the category column as the input and returns the barp
    dataframe: 1-d categorical feature array
    xlabel_df: x axis label
    ylabel_df: y axis label (default 'Count')
    colors: list of colors to use for the different variables
    """
```

```
    # Figure aesthetics
    sns.set_context("talk")

    # Plot informations
    plt.figure(figsize=(12,6))
    plot_df = sns.countplot(dataframe, palette= colors)
    plt.xlabel(xlabel_df)
    plt.ylabel(ylabel_df)

    # Calculating the length of the column
    total = len(dataframe)

    # Looping to calculate percentage of each class of the category and annot
    for cat in plot_df.patches:
        percentage = '{:.1f}%'.format(100 * cat.get_height()/total)

        #setting plot annotate location and size
        x = cat.get_x() + cat.get_width() / 2 - 0.05
        y = cat.get_y() + cat.get_height()
        plot_df.annotate(percentage, (x, y), size = 14)
```

Count of users distributed by Landing Page (New vs Old):

In [18]:
```
# List of colors to use for the different landing page
colors =["peachpuff", "paleturquoise"]

# Using the functio to plot barplot wtih percentage values
bar_perc(news['landing_page'],'landing_page',colors)
```



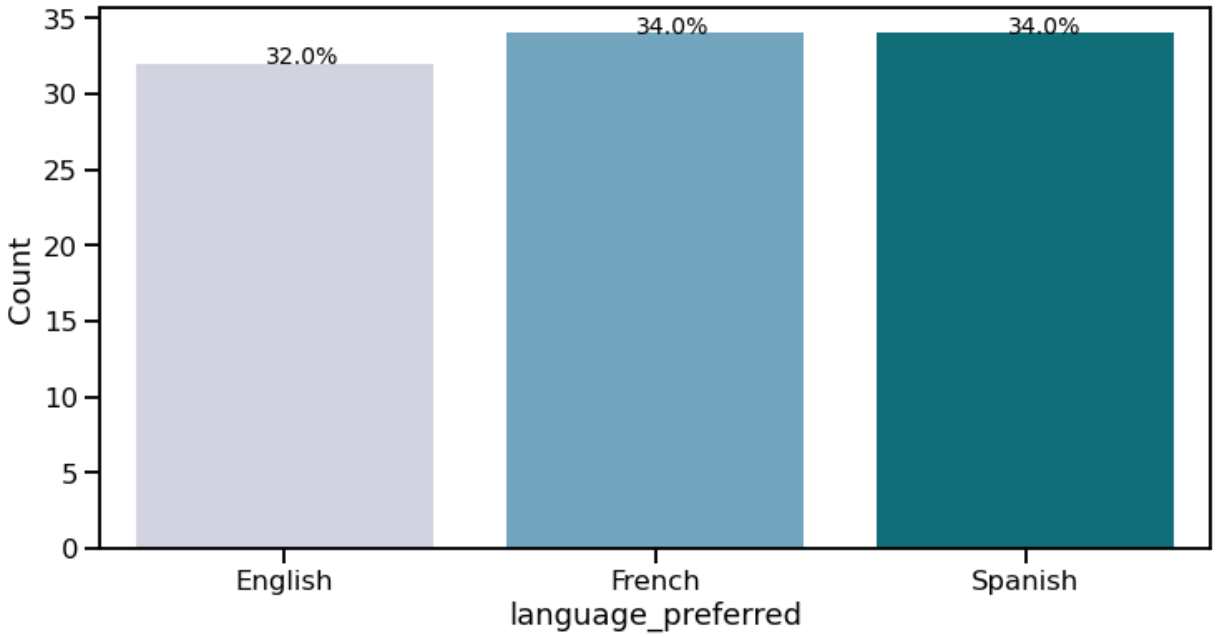
Observations:

- There is 50 users on New landing page (50% of data).
- There is 50 users on Old landing page (50% of data).

Count of users distributed by Language preferred (English vs French vs Spanish):

In [19]:
```
# List of colors to use for the different language
colors ='PuBuGn'
```

```
# Using the functio to plot barplot wtih percentage values
bar_perc(news['language_preferred'],'language_preferred',colors)
```

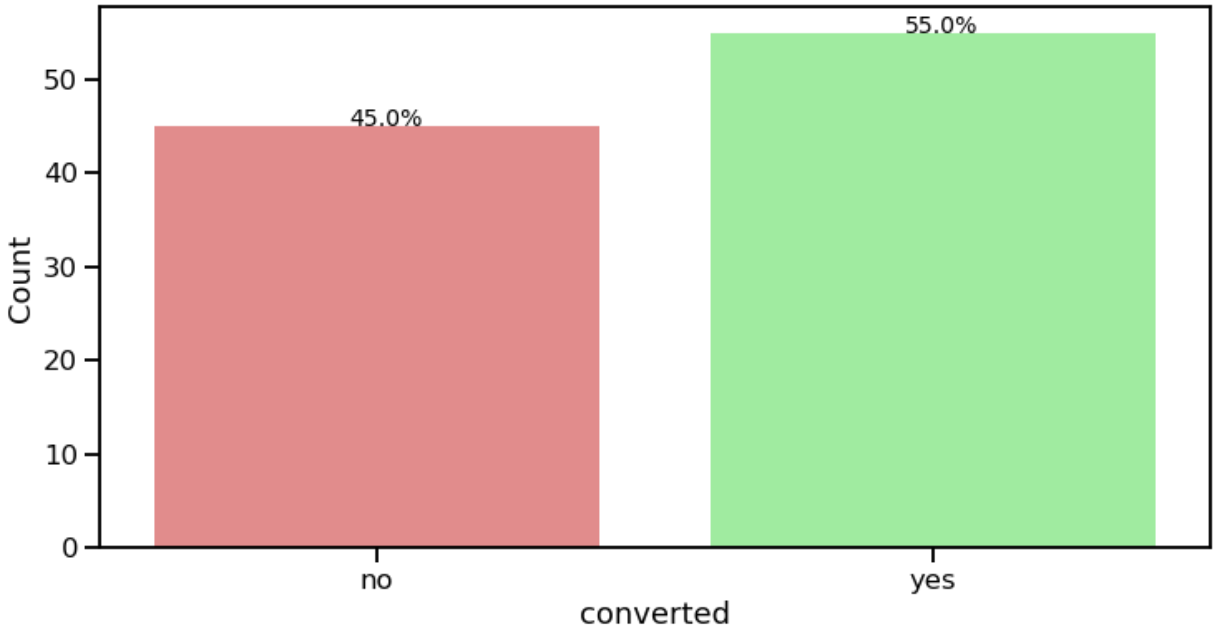


Observations:

- Language preferences its almost equal distributed between the three languages.
- There is 32 users that prefers English as language (32% of data).
- 34 % equal to 34 users prefers to visualize the landing page in French.
- 34 users (34%) use Spanish when landing to the page.

Count of users that got Converted (Became a subscribe? Yes or No):

In [20]:
```
# List of colors to use for the different converted status
colors =['lightcoral','palegreen']

# Using the functio to plot barplot wtih percentage values
bar_perc(news['converted'],'converted',colors)
```



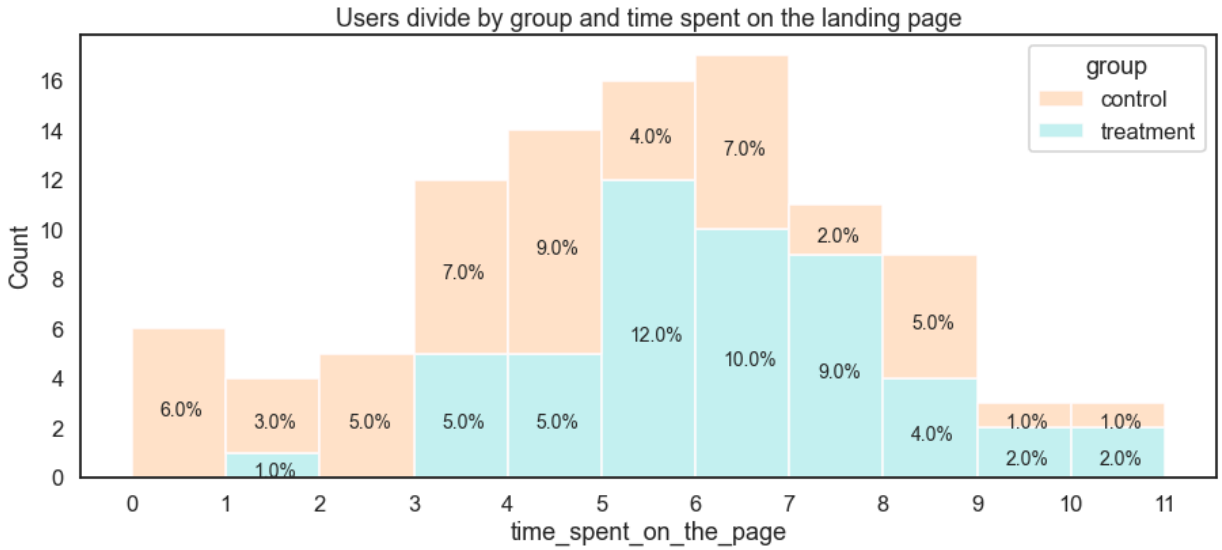
Observations:

- 55% of users became subscribers to the landing page.
- 45% of users do not became subscribers to the landing page.

Bivariate / Multivariate: Understanding relationship between variables

Percentage of users divide by group and time spent on the landing page:

In [21]:
```
#analyzing grorp/landing page distribution by time spent on the page

#Defining plot context parameters.
sns.set_style("white")
plt.figure(figsize=(15,6))
col= ["peachpuff", "paleturquoise"]
sns.set_context("talk")

# Plotting stacked histogram
bins = np.linspace(0,11,12)
plot_group = sns.histplot(data = news,x='time_spent_on_the_page', hue="group"
plt.xticks([0,1,2,3,4,5,6,7,8,9,10,11]) #specifying ticks via the xticks func
plt.title('Users divide by group and time spent on the landing page')

# Calculating the length of the column
total = len(news['time_spent_on_the_page'])

# Looping to calculate percentage of bins and annotate the percentage
for group in plot_group.patches:
    percentage = '{:.1f}%'.format(100 * group.get_height()/total)
    x = group.get_x() + group.get_width() /2 -0.20
    y = group.get_y() + group.get_height() /2 - 0.5
    plot_group.annotate(percentage, (x, y), size = 14)
```



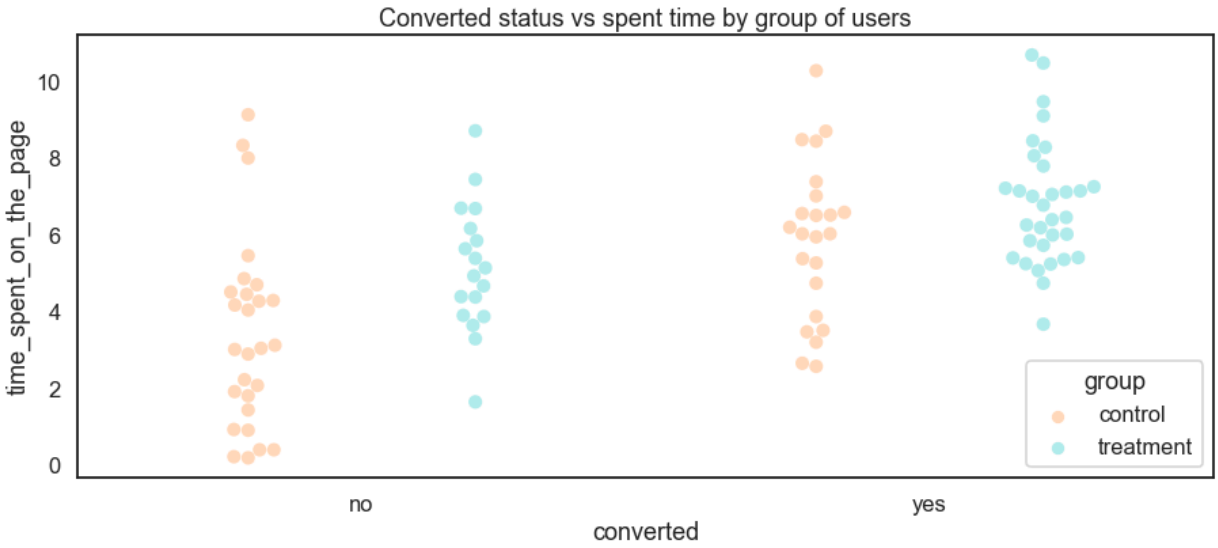
Observations:

- 6% of users spent less than 0 min in the Control/Old page;
- Treatment users spent at least 1 min on the page;
- 8% of Control users spent 1 to 3 min on the page VS 1% of Treatment users;
- 47% of users spent between 4 to 7 min on the page (27% new page/treatment);
- Users on treatment group (New landing page) is more likelihood to spend more time on the page.

Converted status vs spent time on the page by group of users:

In [22]:
```
# Ploting swarmplot for analysis between Converted status vs spent time on th
sns.set_style("white")
plt.figure(figsize=(15,6))
col= ["peachpuff", "paleturquoise"]

sns.swarmplot(x='converted', y='time_spent_on_the_page', data=news, hue=news[

plt.title('Converted status vs spent time by group of users')
```

Out[22]:
```
Text(0.5, 1.0, 'Converted status vs spent time by group of users')
```



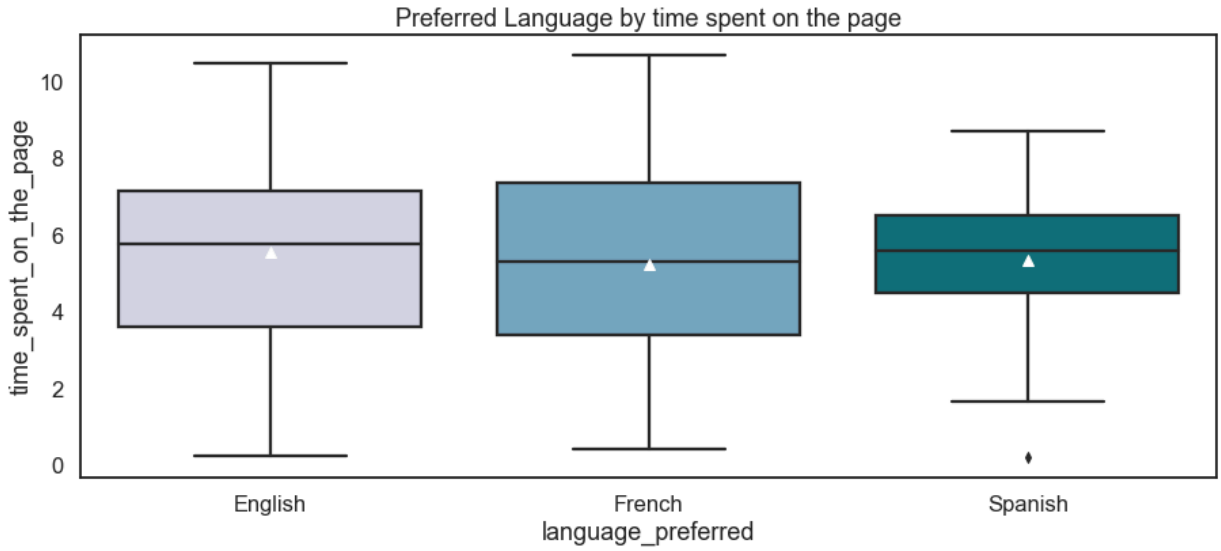
Observations:

- More treatment users / New landing page have been converted.
- Users landing on the new page spent more time on the page compared to old page
- We can see a relation between users that spent more time on the page an got converted

Preferred Language by time spent on the page:

In [23]:
```
# Ploting boxplot for language vs time spent on the page

#Defining plot context parameters.
plt.figure(figsize=(15,6))
sns.set_context("talk")
plt.title('Preferred Language by time spent on the page')

sns.boxplot(news['language_preferred'],news['time_spent_on_the_page'],palette
            meanprops={"markerfacecolor":"white", "markeredgecolor":"white","m
```

Out[23]:
```
<AxesSubplot:title={'center':'Preferred Language by time spent on the page'},
xlabel='language_preferred', ylabel='time_spent_on_the_page'>
```



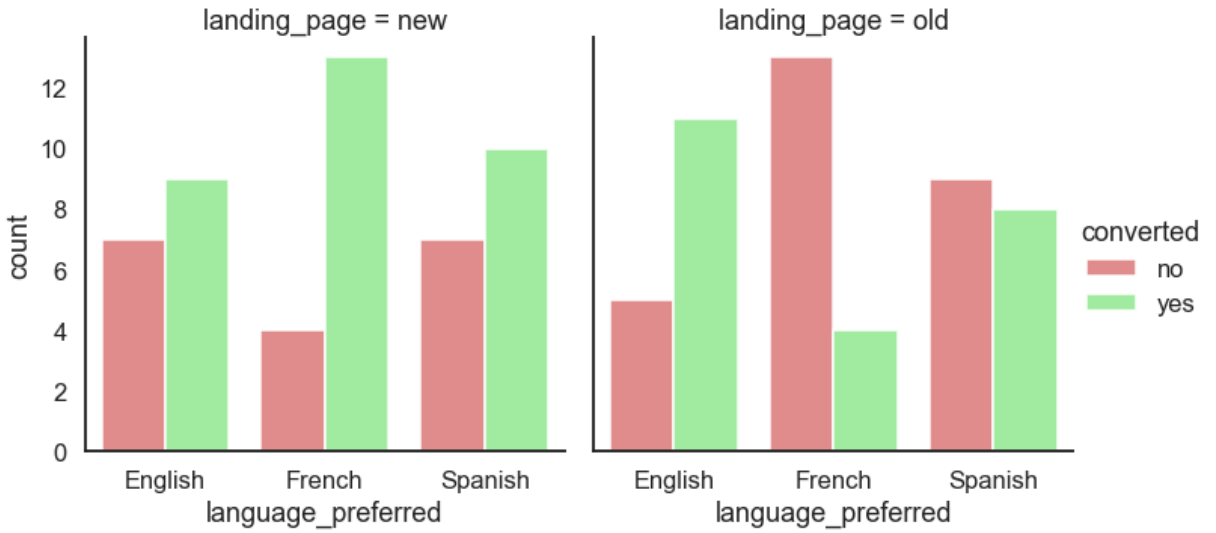
Observations:

- Mean time spent on page is almost equal for all language preferred.
- Spanish users has small variation between time spent on page (concentrated between 4 to 7 min) and, 1 outlier.
- French and English users are more distributed between time spent on the page (greater variation).

Preferred language by Converted status VS landing page

In [24]:
```
# Ploting catplot for language by converted statys VS landing page

plt.figure(figsize=(15,8))
colors =['lightcoral','palegreen']
plot_language=sns.catplot(x="language_preferred", hue="converted", col="landi
                data=news, kind="count", palette= colors)
```

```
<Figure size 1080x576 with 0 Axes>
```



In [25]:
```
# Cross table by columns as landing page, rows with converts and no converts
pd.crosstab(news.converted, [news.language_preferred,news.landing_page], marg
```

Out[25]:

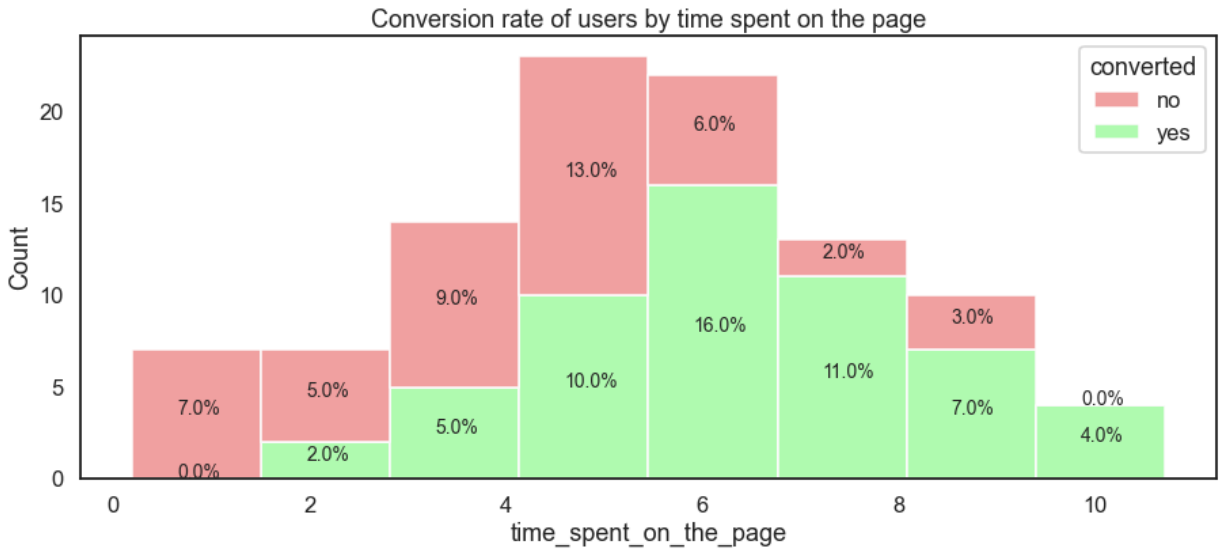
language_preferred	English		French		Spanish		All
landing_page	new	old	new	old	new	old	
converted							
no	7	5	4	13	7	9	45
yes	9	11	13	4	10	8	55
All	16	16	17	17	17	17	100

Observations:

- French users get the opposite conversion status per landing page, showing preference for the New landing page (treatment group).
- English users got more conversion status on the Old landing page (control group). Needs more analysis to understand preference
- Spanish users doesn't show much variation between converts status and landing page; it also needs more analysis to understand preference

Conversion rate of users by time spent on the page

In [26]:
```
#analyzing conversion rate by time spent on the page

sns.set_style("white")
plt.figure(figsize=(15,6))
colors =['lightcoral','palegreen']

# Plotting stacked histogram
plot_time = sns.histplot(data = news,x='time_spent_on_the_page', hue="convert
plt.title('Conversion rate of users by time spent on the page')

# Calculating the length of the column
total = len(news['time_spent_on_the_page'])

# Looping to calculate percentage of bins and annotate the percentage
for time in plot_time.patches:
    percentage = '{:.1f}%'.format(100 * time.get_height()/total)
    x = time.get_x() + time.get_width() /2 -0.20
    y = time.get_y() + time.get_height() /2
    plot_time.annotate(percentage, (x, y), size = 14)
```



Observations:

- Greater the time spent on the page, greater was the number of users converted.

Dataset: Users Time Spent by Landing Page

In [27]:
```
# Creating new dataset, filtering just New page informations
new_page = news[news["landing_page"]=='new']
new_page.head(2)
```

Out[27]:

	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferred
1	546468	treatment	new	7.13	yes	English
2	546462	treatment	new	4.40	no	Spanish

In [28]:
```
# Creating new dataset, filtering just Old page informations
old_page = news[news["landing_page"]=='old']
old_page.head(2)
```

Out[28]:

	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferred
0	546592	control	old	3.48	yes	Spanish
3	546567	control	old	3.02	no	French

EDA and Statistical analysis

Consider a significance level of 0.05 for all tests.

- The sample size , N = 100.
- Users divided equally into two groups (Control = Old page | Tratment = New page).
- Population standard deviation (σ) is unknown.

1. Do the users spend more time on the new landing page than the old landing page?

Step 1: Define null and alternate hypotheses:

> $H_0 : \mu_{new} = \mu_{old}$ (New landing page did not raise the spent time on page)
>
> $H_a : \mu_{new} > \mu_{old}$ (New landing page did raise the spent time on page)

Step 2: Select appropriate Test:

Two sample t-test (independent) - Assumptions:

- Normally distributed
- Random sampling from the population
- Continuous data (spent time on page)
- Independent populations (the two random samples are from two independent populations).
- Unequal population standard deviations (Sample standard deviations are different, the population standard deviations may be assumed to be different).

- Since the sole purpose of the test is to check whether the new landing page is successful compared to old landing page, we would prefer a right tailed t-test.

In [29]:
```
# finding sample means and sample standard deviations for the two samples
print('The mean spent time on the NEW landing page is: ' + str(round(new_page
print('The mean spent time on the OLD landing page is: ' + str(round(old_page
print()
print('The standard deviation of spent time on NEW landing page is ' + str(ro
print('The standard deviation of spent time on OLD landing page is ' + str(ro
```

```
The mean spent time on the NEW landing page is: 6.223
The mean spent time on the OLD landing page is: 4.532

The standard deviation of spent time on NEW landing page is 1.82
The standard deviation of spent time on OLD landing page is 2.58
```

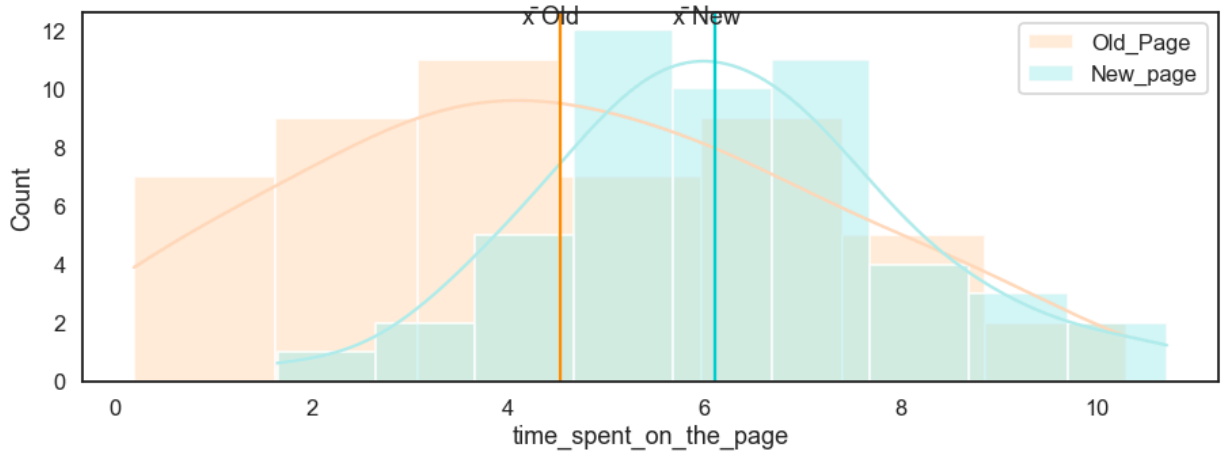
In [30]:
```
#Performing a independence t-test to compare spent time on new vs old landing
t, p_value  = stats.ttest_ind(new_page['time_spent_on_the_page'],old_page['ti
                              alternative = 'greater',equal_var=False)

print(f'test_stats: {t:.4f} \np_value: {p_value:.5f}')
```

```
test_stats: 3.7868
p_value: 0.00014
```

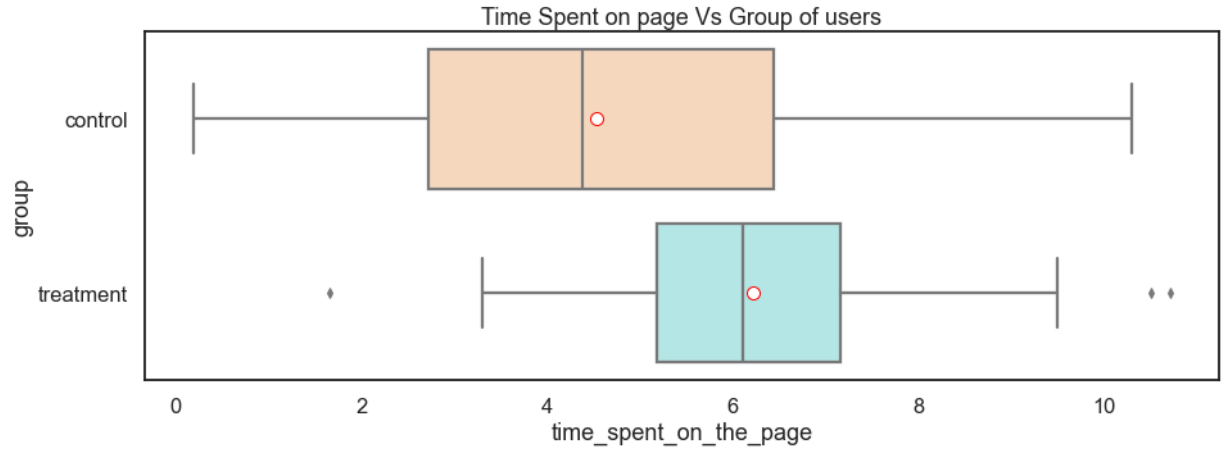
Insight

> As the p-value(~0.00014) is smaller than the level of significance, we have stronge evidence to reject the null hypothesis and conclude that the spent time on the new landing page is greater compared to the old at 0.05 significance level.
>
> There is less than 0.01% of chance of mean spent time on New Landing page to be equal a mean spent time on Old Landing .

In [31]:
```
# Ploting histogram for analysis between time spent on page for each landing
plt.figure(figsize=(15,5))
ttes_hist_old=sns.histplot(old_page['time_spent_on_the_page'],color="peachpuf
ttes_hist_new=sns.histplot(new_page['time_spent_on_the_page'],color="paleturq

# Add mean old page to the histogram
ttes_hist_old.axvline(np.mean(old_page['time_spent_on_the_page']), color='dar
# Add mean new page to the histogram
ttes_hist_new.axvline(np.median(new_page['time_spent_on_the_page']), color='d

plt.annotate('x̄ Old', (4.15, 12.2))
plt.annotate('x̄ New', (5.68, 12.2))

plt.legend(loc='upper right')
plt.show()
```



In [32]:
```
# Ploting boxplot for analysis between time spent on page for each group of u
plt.figure(figsize=(15,5))
col= ["peachpuff", "paleturquoise"]
sns.boxplot(y="group", x='time_spent_on_the_page', data=news, showmeans=True,
            meanprops={"marker":"o","markerfacecolor":"white", "markeredgecol

plt.title('Time Spent on page Vs Group of users')
plt.show()
```



Observations:

- Group control (old page) has mean less than group treatment and data spread between range 0 to 10
- Group treatment (new page) has mean 6.22 minutes per user and some outliers

2. Is the converted rate for new page greater than the conversion rate for the old page?

Step 1: Define null and alternate hypotheses:

> $H_0 : p_{new} = p_{old}$ (Converted rate did not raise for New landing page)
>
> $H_a : p_{new} > p_{old}$ (Converted rate did raise for New landing page)

Step 2: Select appropriate Test:

Two proportion Z-Test - Assumptions:

- Binomially distributed

- Random sampling from the population
- Binomial distribution approximated to normal distribution

The standard thing is to check whether np and n(1-p) are greater than or equal to 10.

$$np_1 = 50 \cdot \frac{32}{50} = 32 \geq 10$$
$$n(1-p_1) = 500 \cdot \frac{50-32}{50} = 18 \geq 10$$
$$np_2 = 50 \cdot \frac{23}{50} = 23 \geq 10$$
$$n(1-p_2) = 50 \cdot \frac{50-23}{50} = 27 \geq 10$$

- Since the sole purpose of the test is to check whether the new landing page converted rate is greater compared to old landing page, we would prefer a right tailed t-test.

In [33]:
```
# Cross table by columns as landing page, rows with converts and no converts
cross2 = pd.crosstab(news.converted, news.landing_page, margins=False)
cross2
```

Out[33]:

landing_page	new	old
converted		
no	18	27
yes	32	23

In [34]:
```
# Set counts of converted users (yes) for each landing page (New vs Old)
new_converted = new_page[new_page['converted']=="yes"]
new = new_converted.converted.count()

old_converted = old_page[old_page['converted']=="yes"]
old = old_converted.converted.count()

conv_counts = np.array([new, old])

# Set sample size for each landing page (New vs Old)
new_samp = new_page.converted.count()
old_samp = old_page.converted.count()

sample = np.array([new_samp,old_samp])
```

In [35]:
```
# Finding p_value applying proportion Z-Test
from statsmodels.stats.proportion import proportions_ztest
z, p_value = proportions_ztest(conv_counts, sample, alternative='larger')

print(f'test_stats: {z:.4f} \np_value: {p_value:.4f}')
```

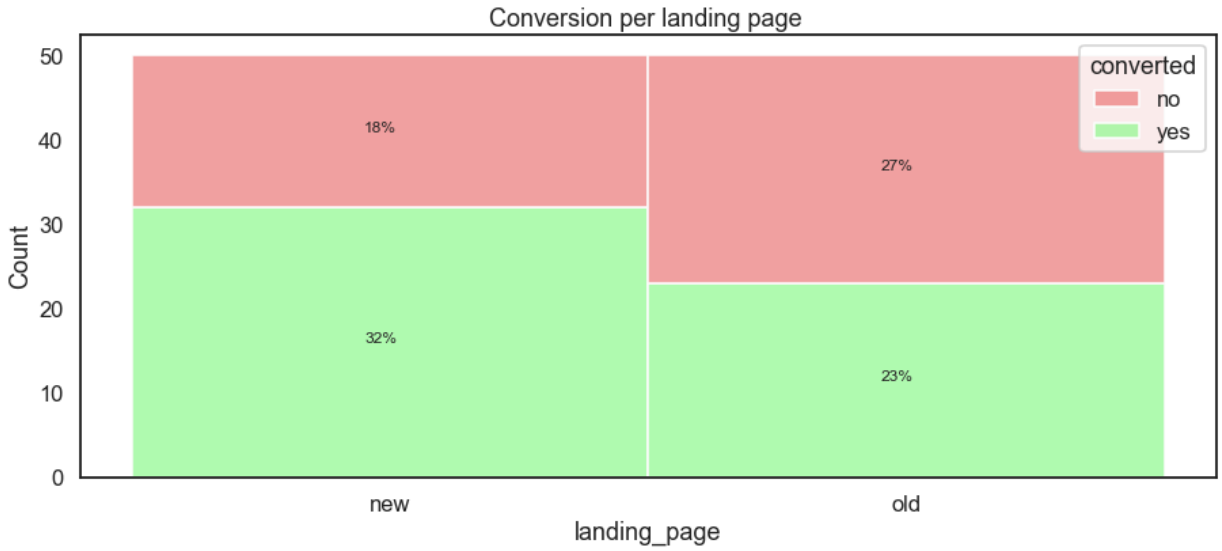
```
test_stats: 1.8091
p_value: 0.0352
```

Insight

> As the p-value(~0.0352) is smaller than the level of significance, we have evidence to reject the null hypothesis and conclude that Converted rate did raise for New landing page at 0.05 significance level.

In [36]:
```
# Analyzing difference on convesion rate between new and old landing page
plt.figure(figsize=(15,6))
```

```
sns.set_style("white")
colors =['lightcoral','palegreen']
plot_landing = sns.histplot(data = news,x='landing_page', hue="converted", mu
plt.title('Conversion per landing page ')

# Calculating the length of the column
total = len(news['landing_page'])

# Looping to calculate percentage of bins and annotate the percentage
for landing in plot_landing.patches:
    percentage = '{:.0f}%'.format(100 * landing.get_height()/total)
    x = landing.get_x() + landing.get_width() /2 -0.05
    y = landing.get_y() + landing.get_height() /2
    plot_landing.annotate(percentage, (x, y), size = 12)
```



Observations:

- More users using new landing page were converted.

3.Does the converted status depend on the preferred language?

Step 1: Define null and alternate hypotheses:

> H_0: Converted status and preferred language are independent
>
> H_a: Converted status and preferred language are not independent

Step 2: Select appropriate Test:

Chi-Square Test for infependence - Assumptions:

- Categorical Variables
- Random sampling from the population
- Expected value of the number of sample observations in each level of the variable is at least 5

In [37]:
```
# Cross table by columns as language, rows with converts and no converts coun
cross = pd.crosstab(news.converted, news.language_preferred, margins=False)
cross
```

Out[37]:

language_preferred	English	French	Spanish
converted			
no	12	17	16
yes	20	17	18

In [38]:
```
# Finding p_value applying Chi-square test of infependence
from scipy.stats import chi2_contingency
chi2, p_value, dof, exp_freq = chi2_contingency(cross)



print(f'The degrees of freedom is: {dof:.0f}')
print(f'The chi2 value is: {chi2:.4f} \nThe p-value is: {p_value:.4f}')
```

```
The degrees of freedom is: 2
The chi2 value is: 1.1289
The p-value is: 0.5687
```

In [39]:
```
print("The expectations of frequence is:")
exp =pd.DataFrame(exp_freq)
exp
```

```
The expectations of frequence is:
```

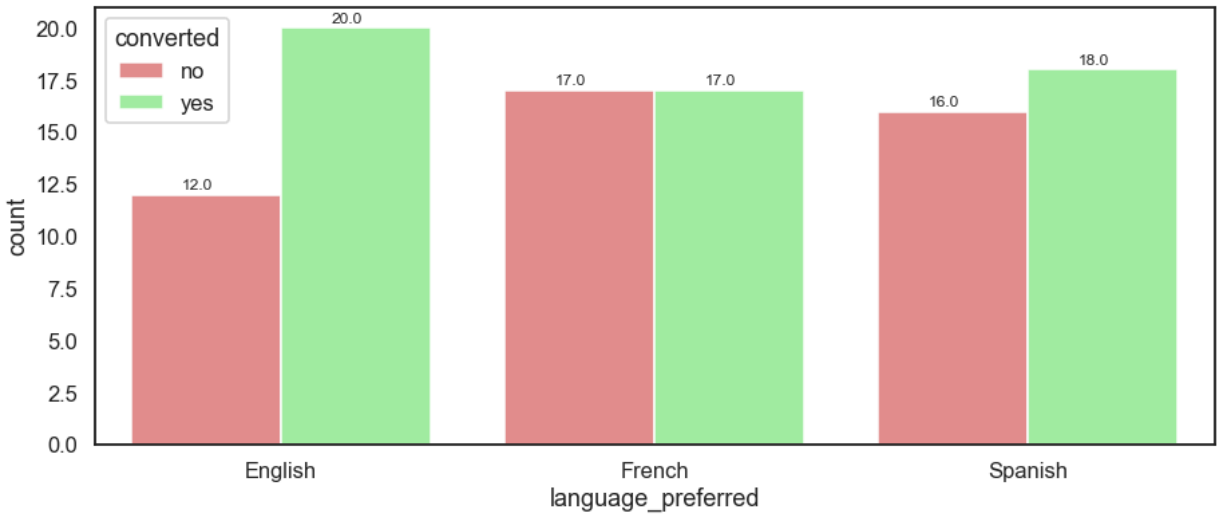
Out[39]:

	0	1	2
0	14.4	15.3	15.3
1	17.6	18.7	18.7

Insight

> A chi-square test of independence was performed to examine the relation between converted status and the user preferred language. The p value is (~0.5686) which is greater than the α 0.05. Hence, we fail to reject the null hypothesis .
>
> This means that we have strong evidence that the converted status and language are independent.

In [40]:
```
plt.figure(figsize=(15,6))
colors =['lightcoral','palegreen']
plot_language = sns.countplot(x="language_preferred", hue="converted", data=n

# Calculating the length of the column
total = len(news['language_preferred'])

# Looping to calculate percentage of bins and annotate the percentage
for language in plot_language.patches:
    count = (100 * language.get_height()/total)
    x = language.get_x() + language.get_width() /3
    y = language.get_y() + language.get_height()+0.25
    plot_language.annotate(count, (x, y), size = 12)
```



4- Is the mean time spent on the new page same for the different language users?

Step 1: Define null and alternate hypotheses:

> $H_0 : \mu_{English} = \mu_{Spanish} = \mu_{French}$
>
> H_a: At least one of these means is not the same

Step 2: Select appropriate Test:

Analysis of variance (ANOVA) - Assumptions:

- Normally distribution
- Random sampling from the population
- Population standard deviation equal.

In [41]:
```
# data from New page users
new_page

# perform one-way anova test, filtering time spent on the page by language pr
f_ratio_value, p_value = stats.f_oneway(new_page.loc[new_page['language_prefe
                                new_page.loc[new_page['language_preferred'] ==
                                new_page.loc[new_page['language_preferred'] ==
print(f'The f_ratio_value is {f_ratio_value:.4f}')
print(f'The p-value is {p_value:.4f}')
```

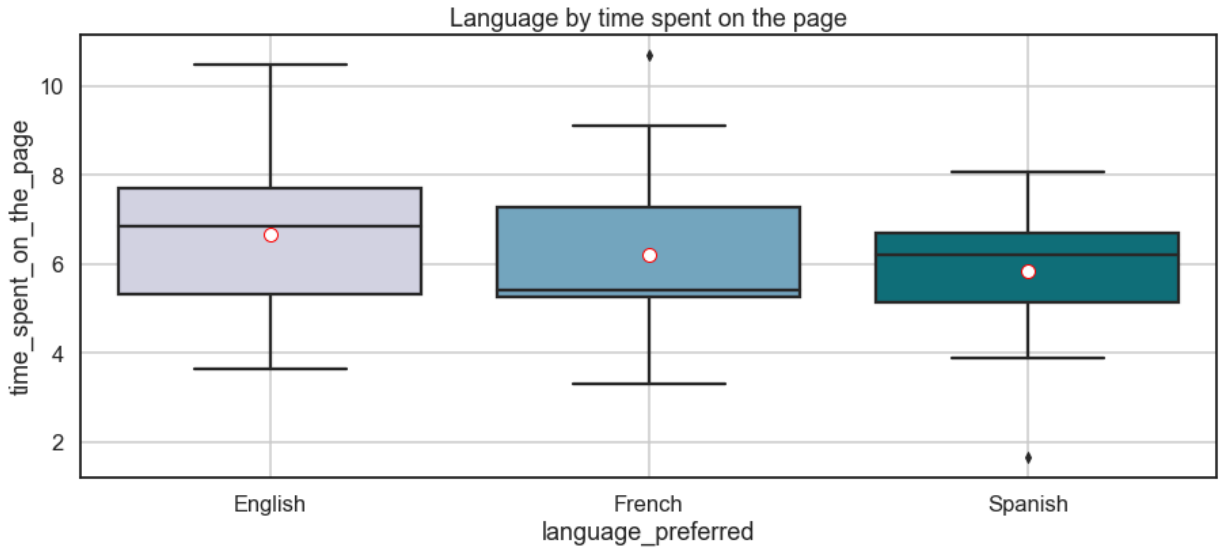
```
The f_ratio_value is 0.8544
The p-value is 0.4320
```

Insight

> In this scenario, the p value is (~0.4320) which is greater than the α 0.05.
> Hence, we fail to reject the null hypothesis .
>
> *This means that we have strong evidence that the mean time spent on the new page are the same for different language users.*

In [42]:
```
# Ploting boxplot for language by time spent on the page
plt.figure(figsize=(15,6))
```

```
sns.boxplot(new_page['language_preferred'],new_page['time_spent_on_the_page']
            meanprops={"marker":"o","markerfacecolor":"white", "markeredgecol

plt.title('Language by time spent on the page')
plt.grid()
```



Business Analysis:

Insights

- Users of the New landing page usually spent more time on the page.
- Control/Old page: 6% of users spent less than 1 min on the page, and 8 % of users less than 3 min.
- Treatment/New page: 1% of users spent between 1 to 2 min and other users spent more than 3min
- New landing page (treatment group), has a greater converted rate compare to the old page.
- There is a relation between users that spent more time on the page an got converted (slide6/7).
- Language preferences its almost equal distributed between the two types of pages.
- Converted status is independent of language preferred;
- The mean spent time on page doesn't change for users with different language preference.

Recommendations for further investigate:

- Deep analyze on the relationship between language preferred Vs preference landing page (Slide8):
- New landing page and Old landing page has similar numbers of users preferred language, with different converted preference (French seems to like New landing page, English seems to like the Old)
- Users who spent more time on page, got converted, we should analyze if this users after became subscribers, if they change the spent time (Paired T-test, same users, before

and after subscribe).

- Why does New Landing page make users spent at least more than 1 min on the page?
- What does make users spent more time on the page?
- Analyze profile of users that spent more than average on the page.

In [ ]: