In [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # Data Visualization
import seaborn as sns # Data Visualization
```

In [2]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
df = pd.read_csv('collegePlace.csv')
```

In [4]:

```
df.head()
```

Out[4]:

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	Male	Electronics And Communication	1	8	1	1	1
1	21	Female	Computer Science	0	7	1	1	1
2	22	Female	Information Technology	1	6	0	0	1
3	21	Male	Information Technology	0	8	0	1	1
4	22	Male	Mechanical	0	8	1	0	1

In [5]:

```
df.tail()
```

Out[5]:

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
2961	23	Male	Information Technology	0	7	0	0	0
2962	23	Male	Mechanical	1	7	1	0	0
2963	22	Male	Information Technology	1	7	0	0	0
2964	22	Male	Computer Science	1	7	0	0	0
2965	23	Male	Civil	0	8	0	0	1

In [6]:

```
df.shape
```

Out[6]:

```
(2966, 8)
```

In [7]:

```
df.columns
```

Out[7]:

```
Index(['Age', 'Gender', 'Stream', 'Internships', 'CGPA', 'Hostel',
       'HistoryOfBacklogs', 'PlacedOrNot'],
      dtype='object')
```

In [8]:

```
df.duplicated().sum()
```

Out[8]:

```
1829
```

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
Age                  0
Gender               0
Stream               0
Internships          0
CGPA                 0
Hostel               0
HistoryOfBacklogs    0
PlacedOrNot          0
dtype: int64
```

In [10]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2966 entries, 0 to 2965
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Age                2966 non-null   int64 
 1   Gender             2966 non-null   object
 2   Stream             2966 non-null   object
 3   Internships        2966 non-null   int64 
 4   CGPA               2966 non-null   int64 
 5   Hostel             2966 non-null   int64 
 6   HistoryOfBacklogs  2966 non-null   int64 
 7   PlacedOrNot        2966 non-null   int64 
dtypes: int64(6), object(2)
memory usage: 185.5+ KB
```

In [11]:

```
df.describe()
```

Out[11]:

	Age	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
count	2966.000000	2966.000000	2966.000000	2966.000000	2966.000000	2966.000000
mean	21.485840	0.703641	7.073837	0.269049	0.192178	0.552596
std	1.324933	0.740197	0.967748	0.443540	0.394079	0.497310
min	19.000000	0.000000	5.000000	0.000000	0.000000	0.000000
25%	21.000000	0.000000	6.000000	0.000000	0.000000	0.000000
50%	21.000000	1.000000	7.000000	0.000000	0.000000	1.000000
75%	22.000000	1.000000	8.000000	1.000000	0.000000	1.000000
max	30.000000	3.000000	9.000000	1.000000	1.000000	1.000000

In [12]:

```
df.nunique()
```

Out[12]:

```
Age                  11
Gender                2
Stream                6
Internships           4
CGPA                  5
Hostel                2
HistoryOfBacklogs     2
PlacedOrNot           2
dtype: int64
```

In [13]:

```
df['Age'].unique()
```

Out[13]:

```
array([22, 21, 23, 24, 28, 30, 25, 26, 20, 19, 29], dtype=int64)
```

In [14]:

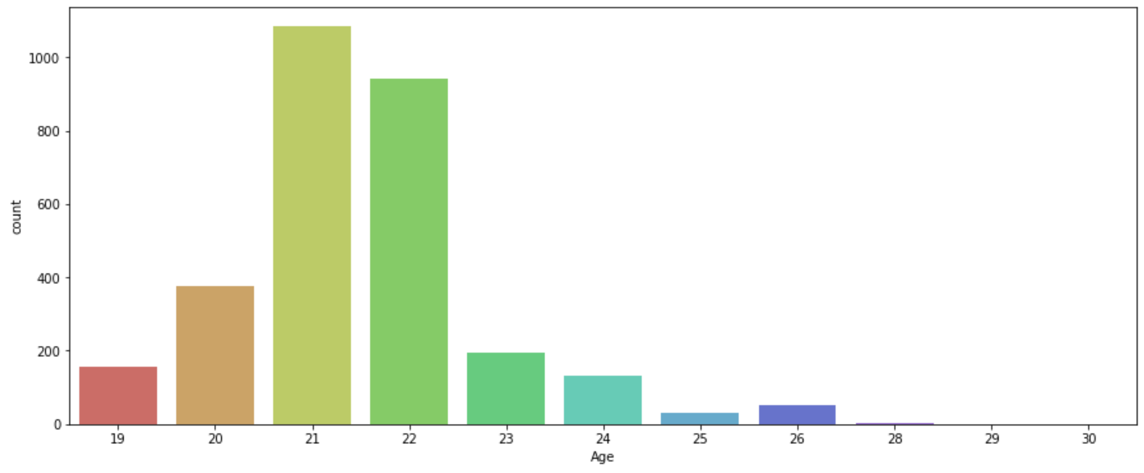
```
df['Age'].value_counts()
```

Out[14]:

```
21    1084
22     941
20     375
23     195
19     156
24     131
26      50
25      29
28       3
30       1
29       1
Name: Age, dtype: int64
```

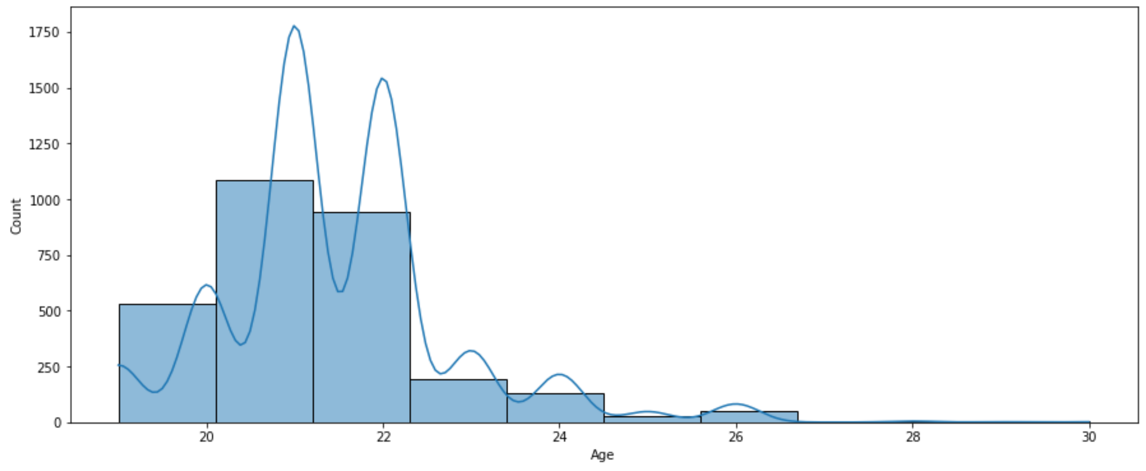
In [15]:

```
plt.figure(figsize=(15,6))
sns.countplot('Age', data = df, palette = 'hls')
plt.show()
```



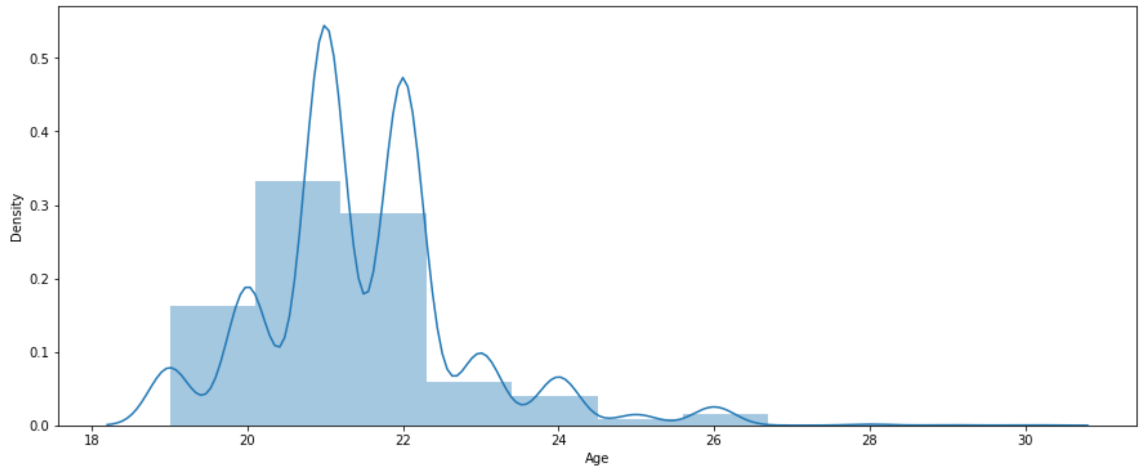
In [16]:

```
plt.figure(figsize=(15,6))
sns.histplot(df['Age'], kde = True, bins = 10, palette = 'hls')
plt.show()
```



In [17]:

```
plt.figure(figsize=(15,6))
sns.distplot(df['Age'], kde = True, bins = 10)
plt.show()
```



In [18]:

```
df['Gender'].unique()
```

Out[18]:

```
array(['Male', 'Female'], dtype=object)
```

In [19]:

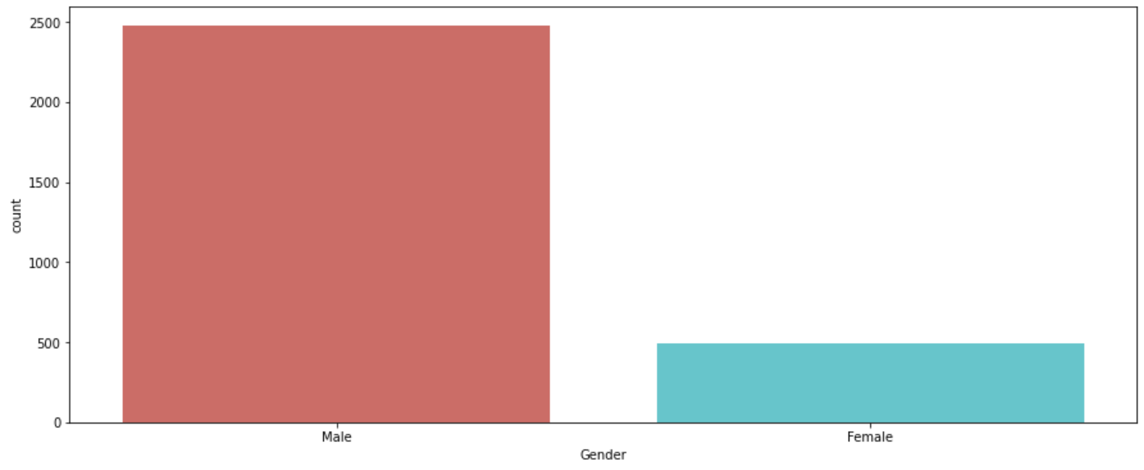
```
df['Gender'].value_counts()
```

Out[19]:

```
Male      2475
Female     491
Name: Gender, dtype: int64
```

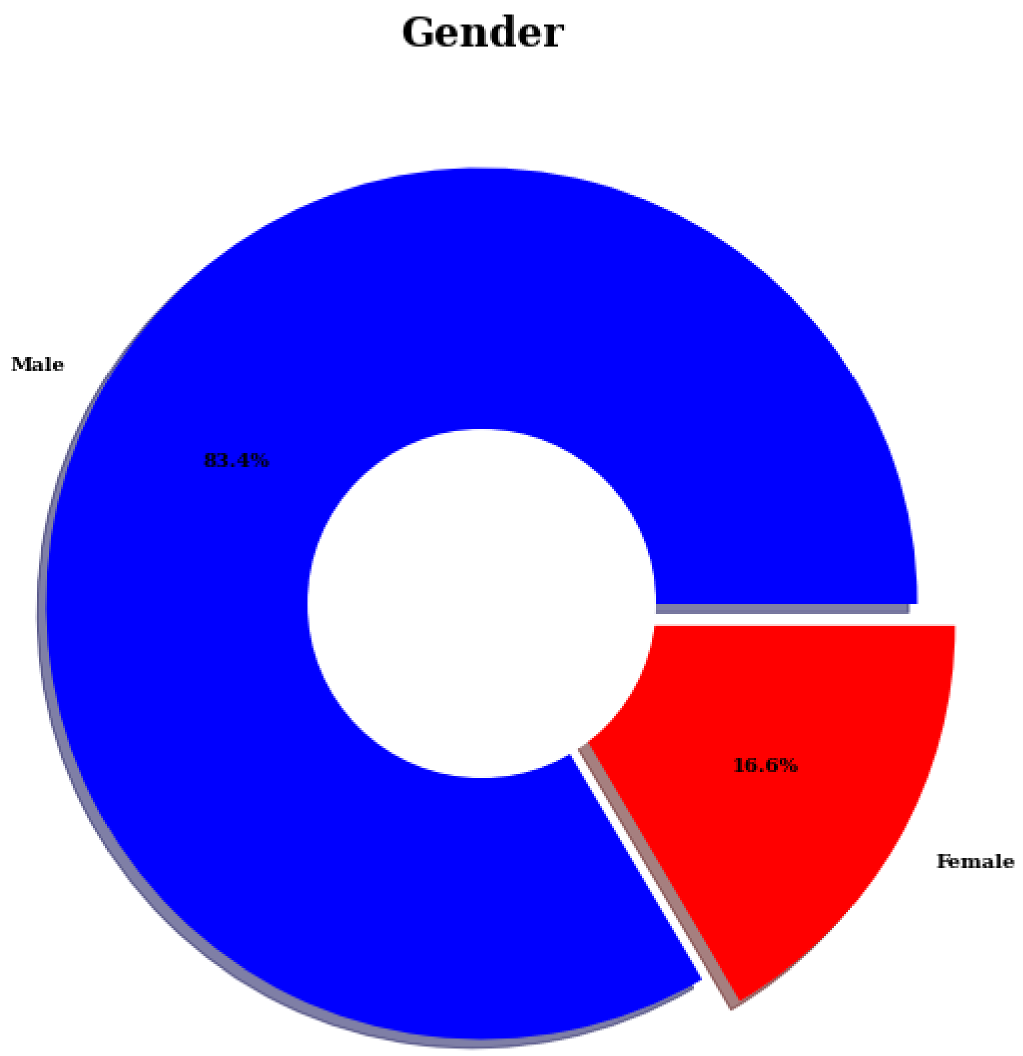
In [20]:

```
plt.figure(figsize=(15,6))
sns.countplot('Gender', data = df, palette = 'hls')
plt.show()
```



In [21]:

```
label_data = df['Gender'].value_counts()

explode = (0.0, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Gender', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [22]:

```
df.Stream.unique()
```

Out[22]:

```
array(['Electronics And Communication', 'Computer Science',
       'Information Technology', 'Mechanical', 'Electrical', 'Civil'],
      dtype=object)
```

In [23]:

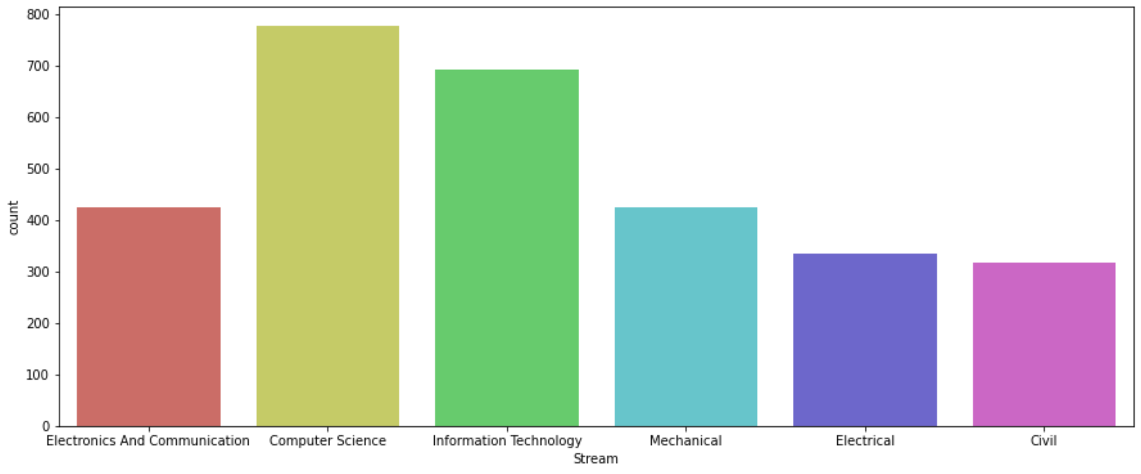
```
df.Stream.value_counts()
```

Out[23]:

```
Computer Science                 776
Information Technology           691
Electronics And Communication    424
Mechanical                       424
Electrical                       334
Civil                            317
Name: Stream, dtype: int64
```

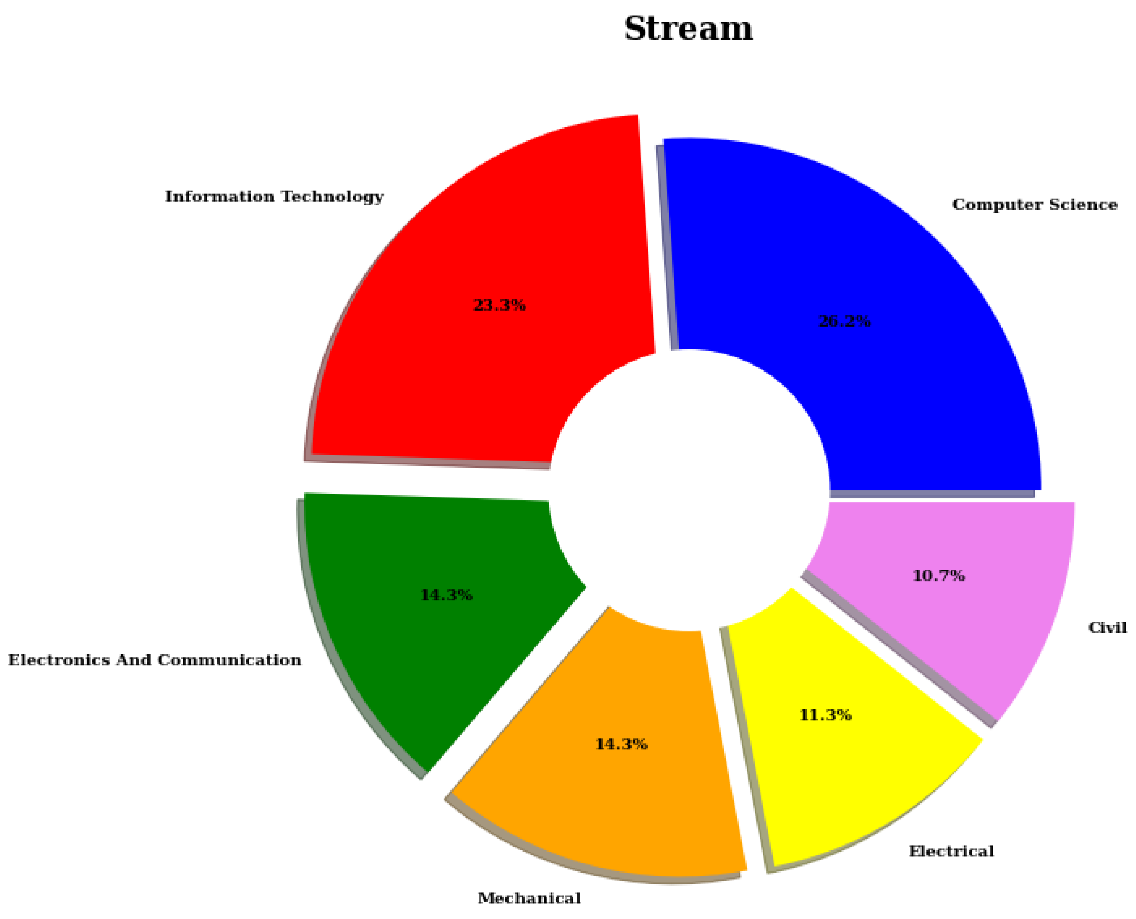
In [24]:

```
plt.figure(figsize=(15,6))
sns.countplot('Stream', data = df, palette = 'hls')
plt.show()
```



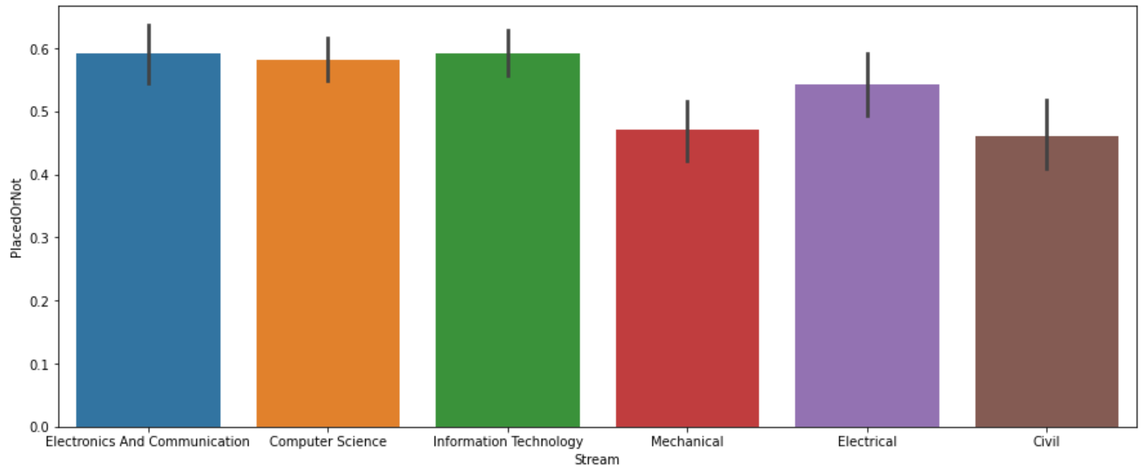
In [25]:

```
label_data = df['Stream'].value_counts()

explode = (0.0, 0.1, 0.1, 0.1, 0.1, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red', 'green', 'orange', 'yellow', 'vi
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Stream', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



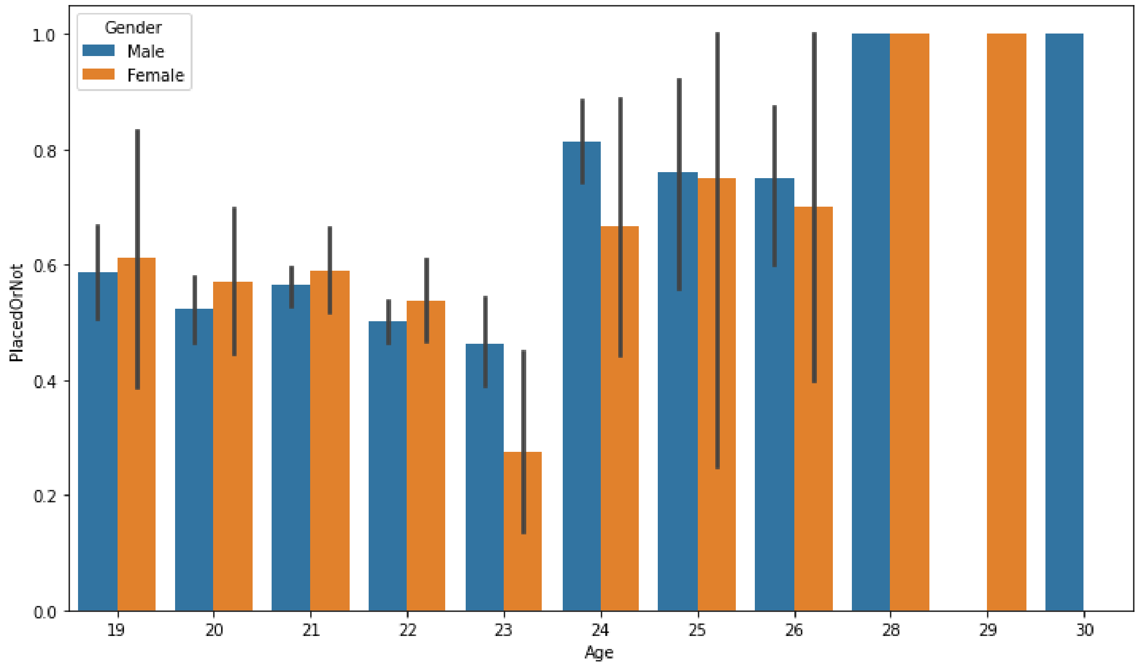
In [26]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df.Stream, y = df.PlacedOrNot)
plt.show()
```



In [27]:

```
plt.figure(figsize = (12,7))
sns.barplot(x = df.Age, y = df.PlacedOrNot, hue = df.Gender)
plt.show()
```



In [28]:

```
df['Internships'].unique()
```

Out[28]:

```
array([1, 0, 2, 3], dtype=int64)
```

In [29]:

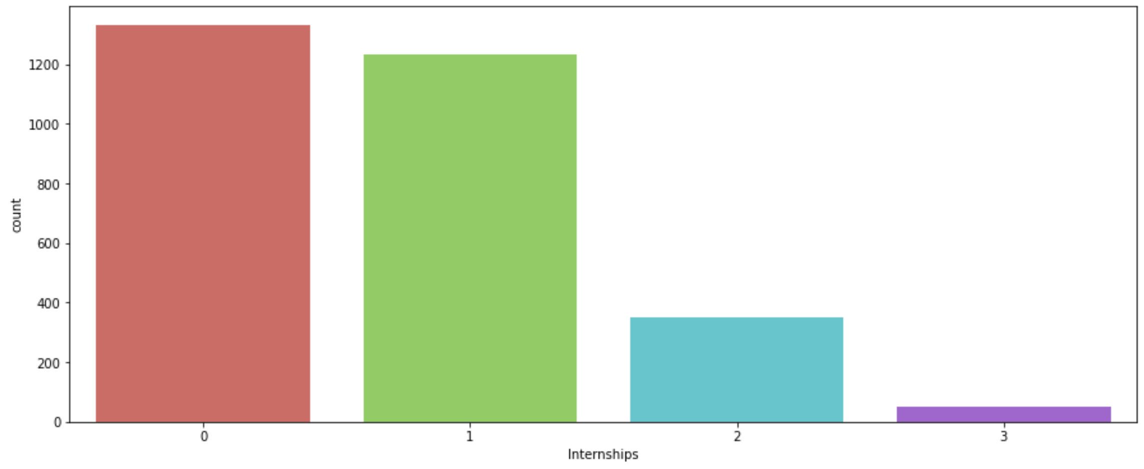
```
df['Internships'].value_counts()
```

Out[29]:

```
0    1331
1    1234
2     350
3      51
Name: Internships, dtype: int64
```

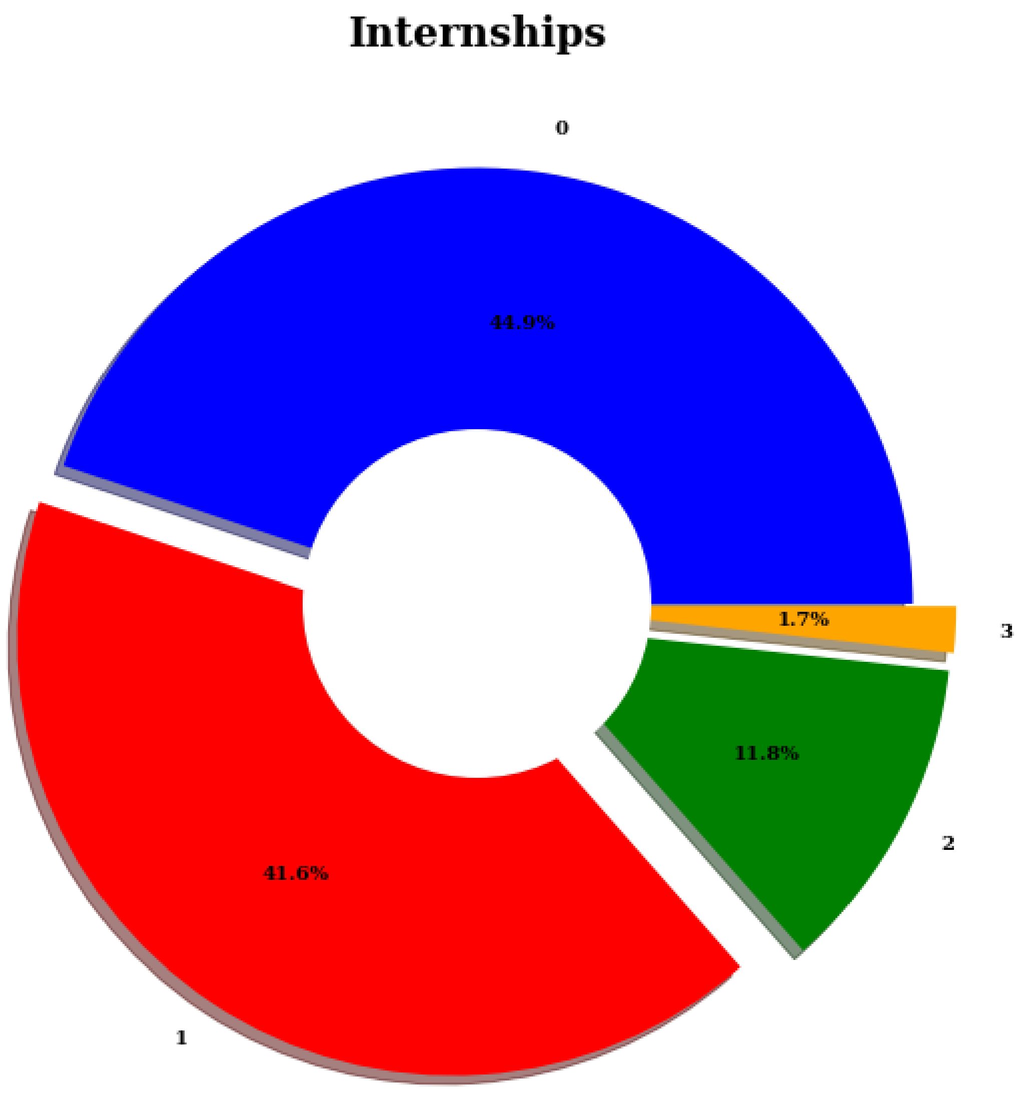
In [30]:

```
plt.figure(figsize=(15,6))
sns.countplot('Internships', data = df, palette = 'hls')
plt.show()
```



In [31]:

```
label_data = df['Internships'].value_counts()

explode = (0.0, 0.1, 0.1, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red', 'green', 'orange'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Internships', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [32]:

```
df['CGPA'].unique()
```

Out[32]:

```
array([8, 7, 6, 9, 5], dtype=int64)
```

In [33]:

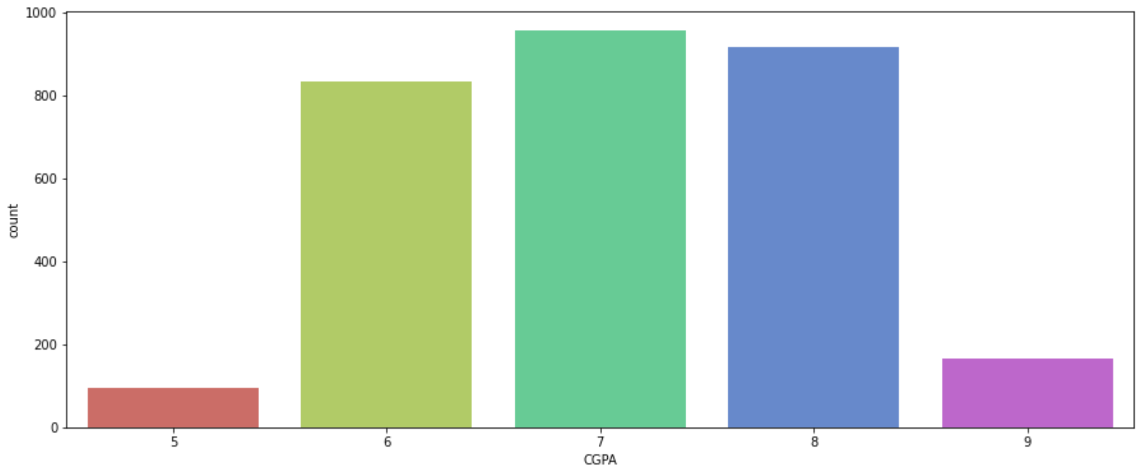
```
df['CGPA'].value_counts()
```

Out[33]:

```
7    956
8    915
6    834
9    165
5     96
Name: CGPA, dtype: int64
```

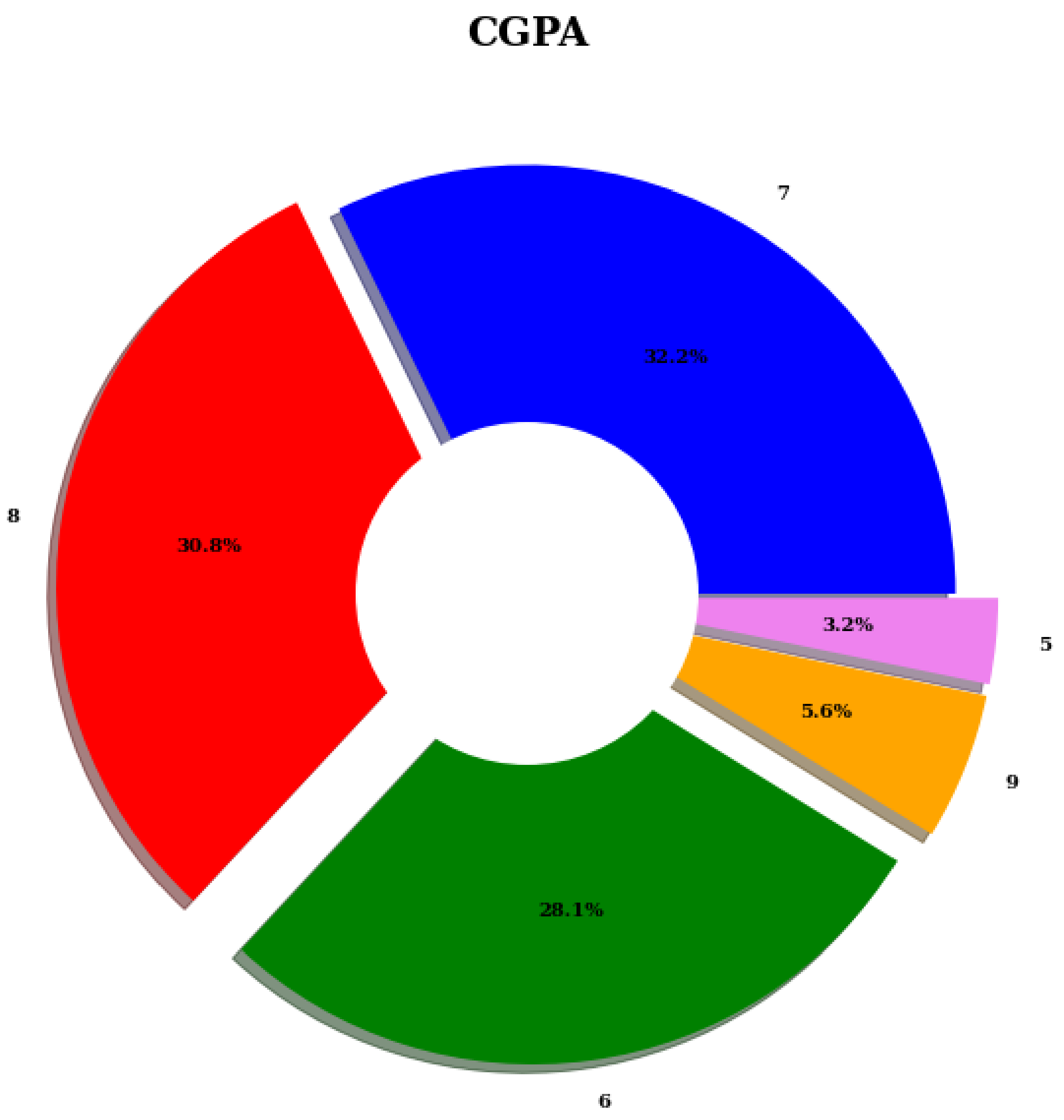
In [34]:

```
plt.figure(figsize=(15,6))
sns.countplot('CGPA', data = df, palette = 'hls')
plt.show()
```



In [35]:

```
label_data = df['CGPA'].value_counts()

explode = (0.0, 0.1, 0.1, 0.1, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red', 'green', 'orange', 'violet'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('CGPA', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [36]:

```
df['Hostel'].unique()
```

Out[36]:

```
array([1, 0], dtype=int64)
```

In [37]:

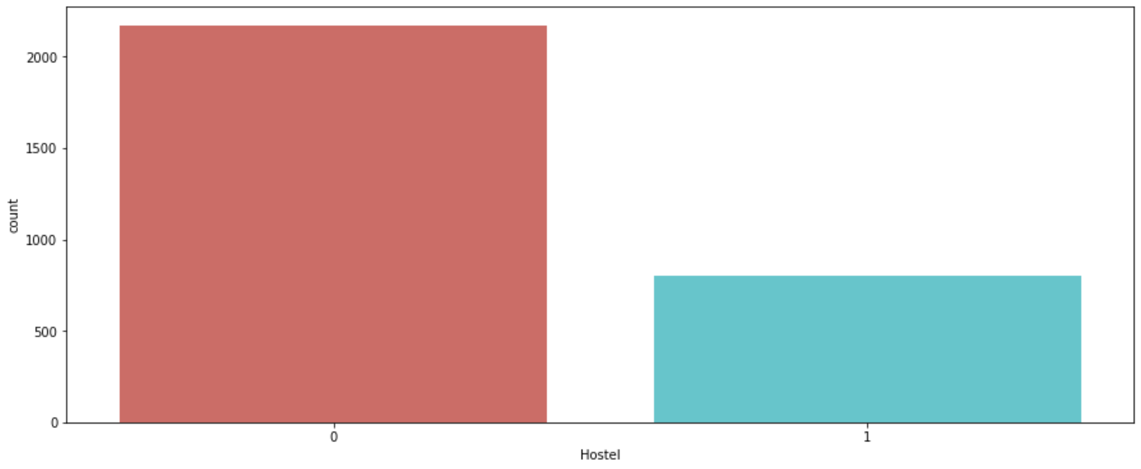
```
df['Hostel'].value_counts()
```

Out[37]:

```
0    2168
1     798
Name: Hostel, dtype: int64
```

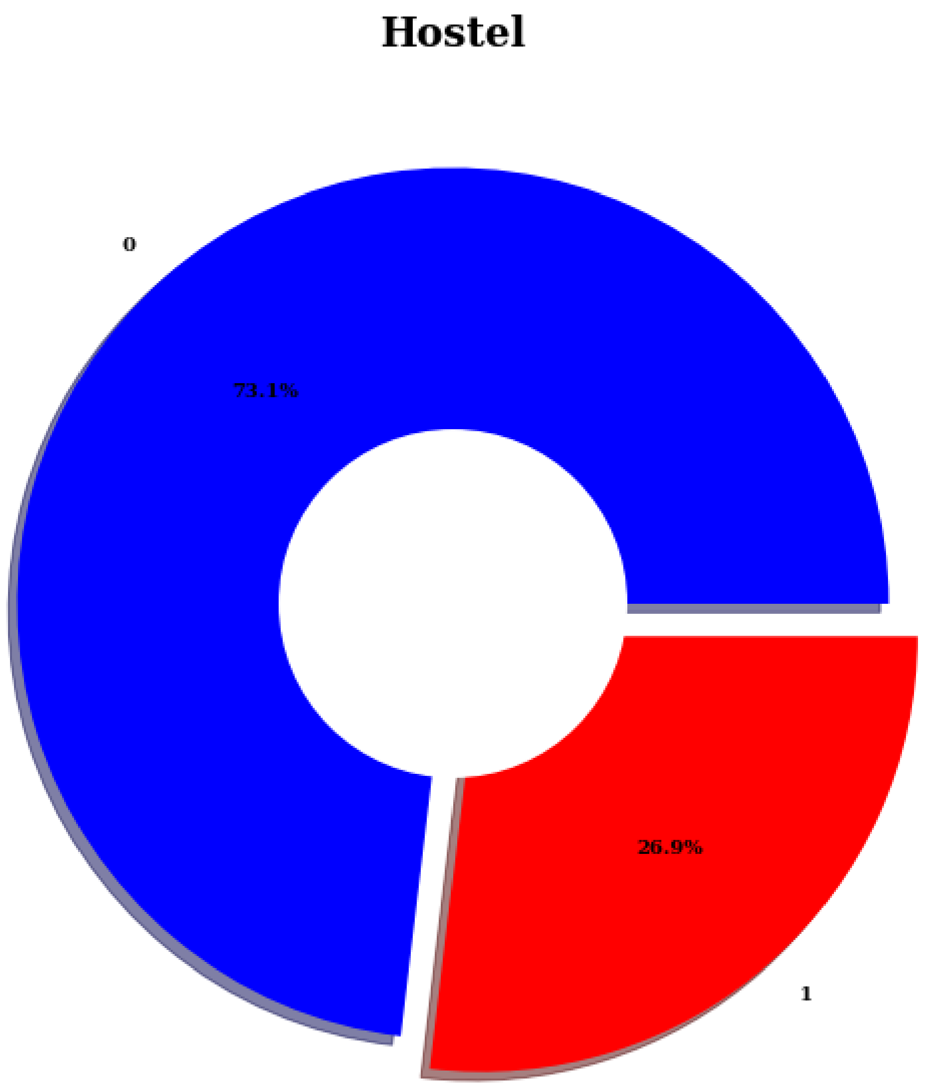
In [38]:

```
plt.figure(figsize=(15,6))
sns.countplot('Hostel', data = df, palette = 'hls')
plt.show()
```



In [39]:

```
label_data = df['Hostel'].value_counts()

explode = (0.0, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Hostel', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [40]:

```
df['HistoryOfBacklogs'].unique()
```

Out[40]:

```
array([1, 0], dtype=int64)
```

In [41]:

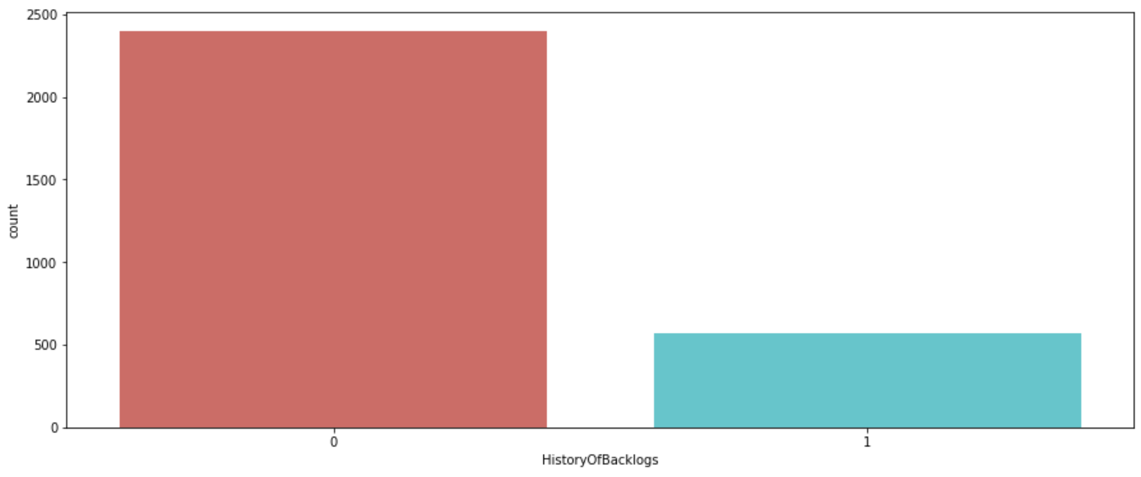
```
df['HistoryOfBacklogs'].value_counts()
```

Out[41]:

```
0    2396
1     570
Name: HistoryOfBacklogs, dtype: int64
```

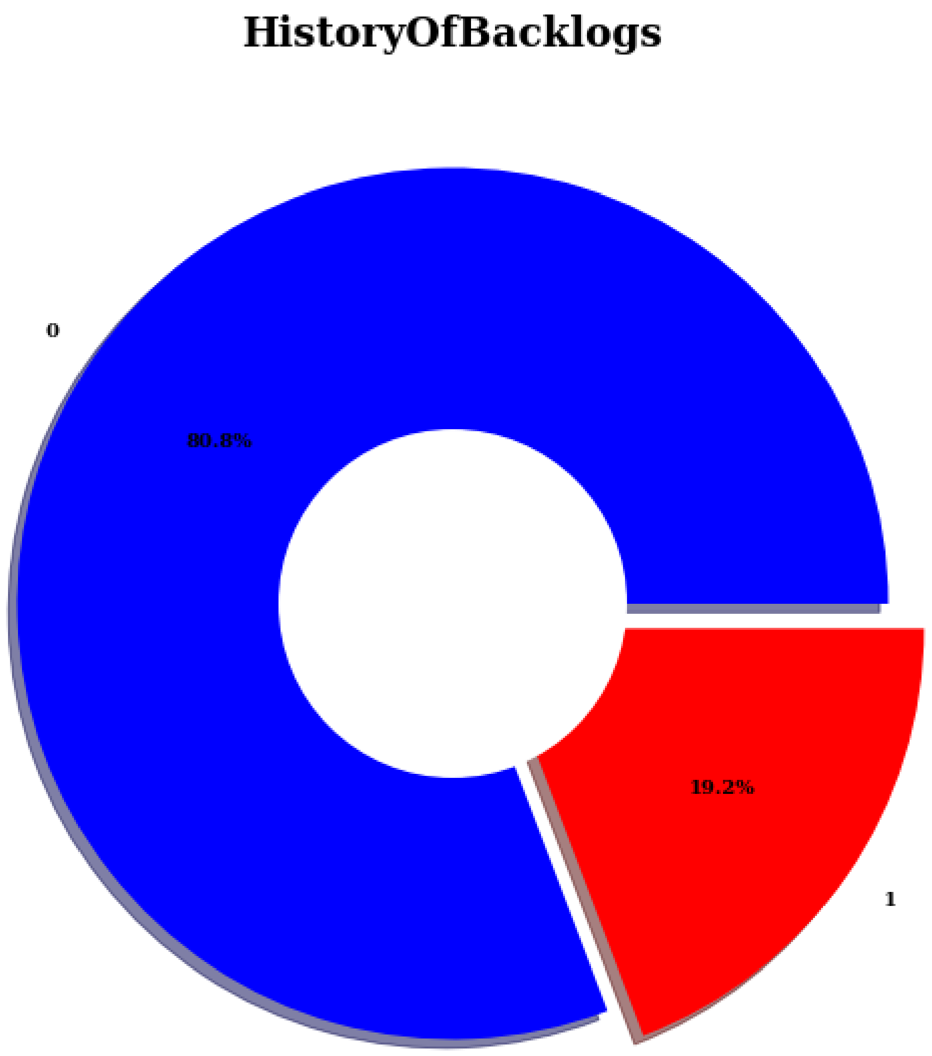
In [42]:

```
plt.figure(figsize=(15,6))
sns.countplot('HistoryOfBacklogs', data = df, palette = 'hls')
plt.show()
```



In [43]:

```
label_data = df['HistoryOfBacklogs'].value_counts()

explode = (0.0, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('HistoryOfBacklogs', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [44]:

```
df['PlacedOrNot'].unique()
```

Out[44]:

```
array([1, 0], dtype=int64)
```

In [45]:

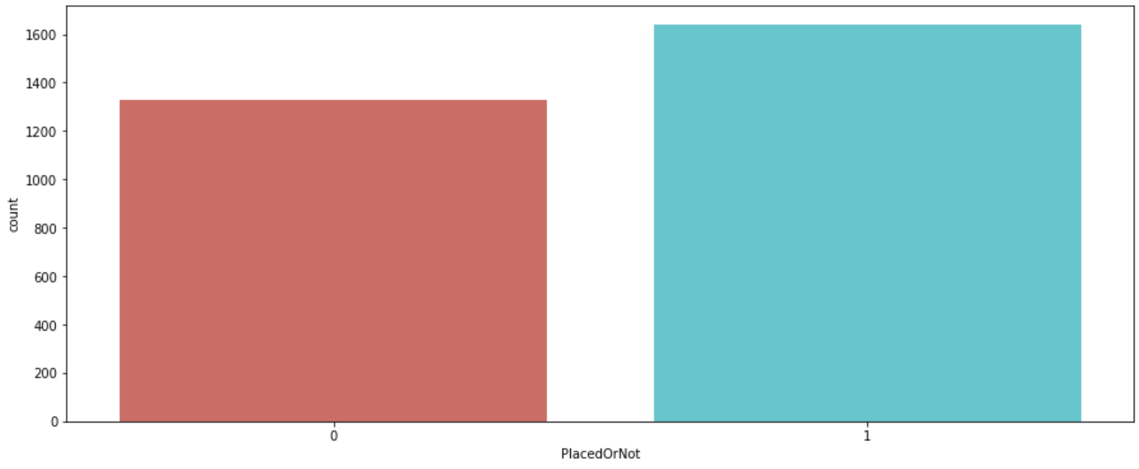
```
df['PlacedOrNot'].value_counts()
```

Out[45]:

```
1    1639
0    1327
Name: PlacedOrNot, dtype: int64
```

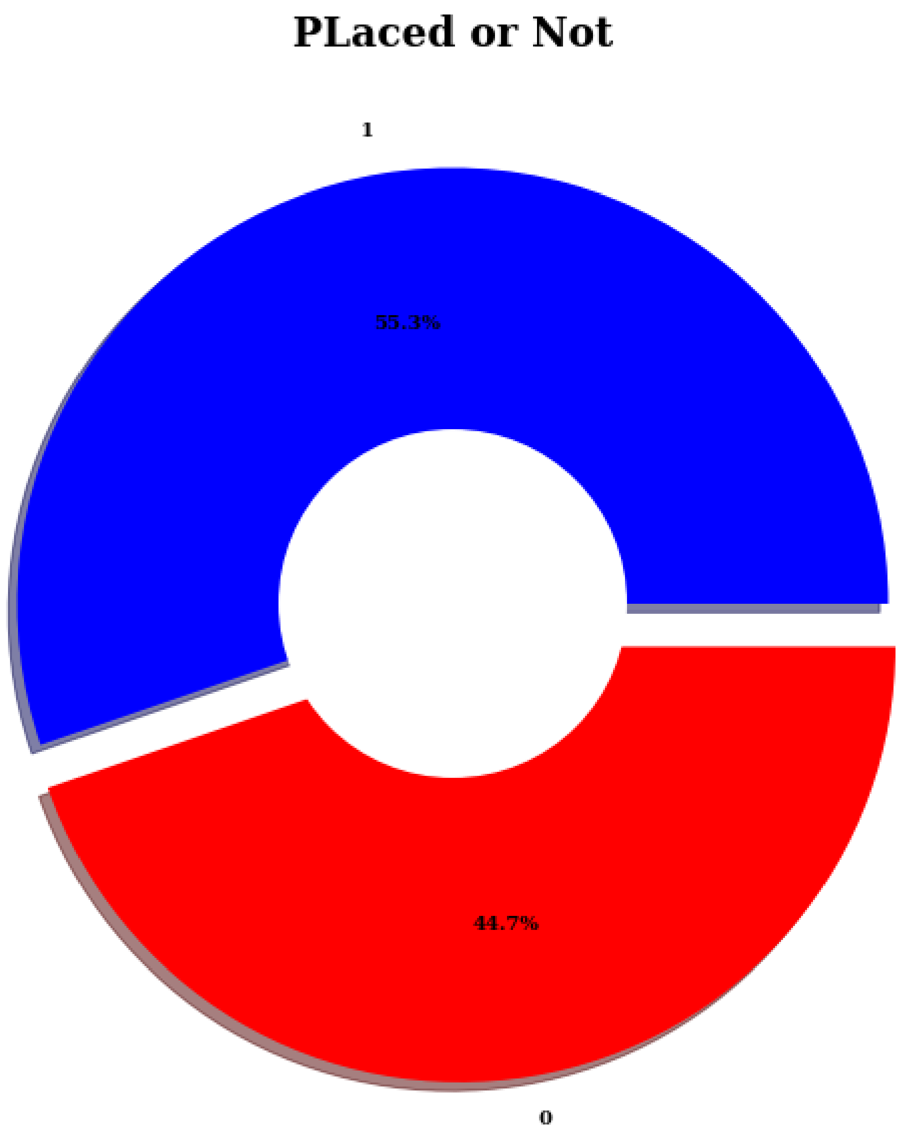
In [46]:

```python
plt.figure(figsize=(15,6))
sns.countplot('PlacedOrNot', data = df, palette = 'hls')
plt.show()
```



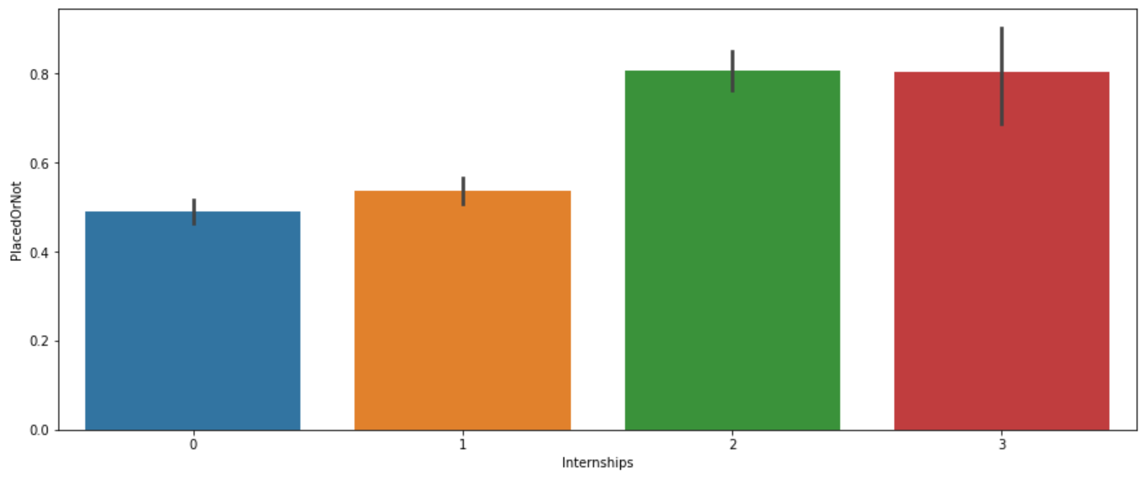
In [47]:

```
label_data = df['PlacedOrNot'].value_counts()

explode = (0.0, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                               labels = label_data.index,
                               colors = ['blue', 'red'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 0,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 10,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('PLaced or Not', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



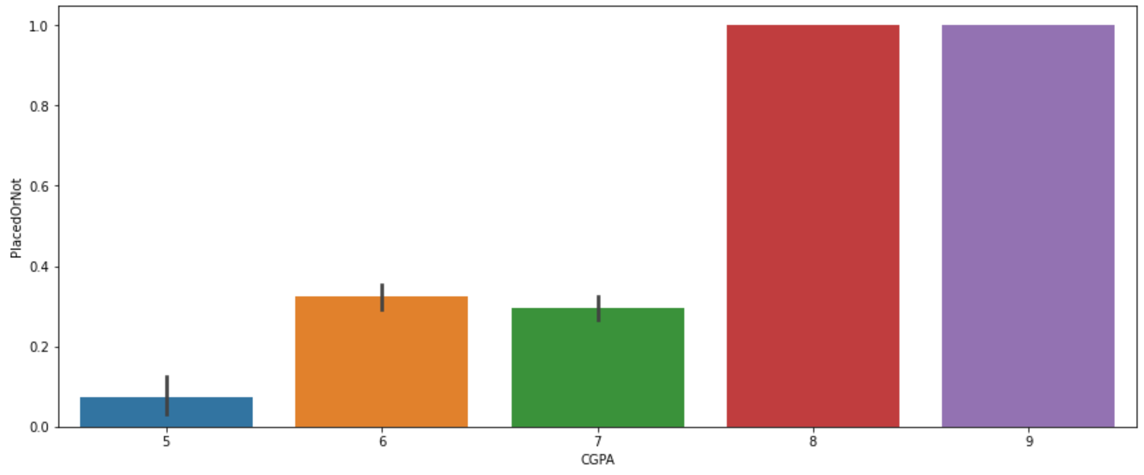
In [48]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df.Internships, y = df.PlacedOrNot)
plt.show()
```



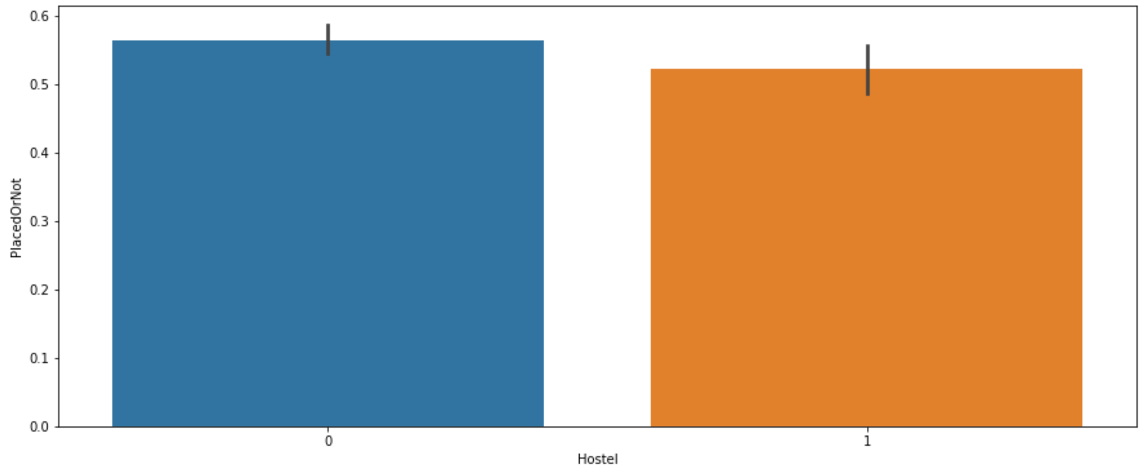
In [49]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df.CGPA, y = df.PlacedOrNot)
plt.show()
```



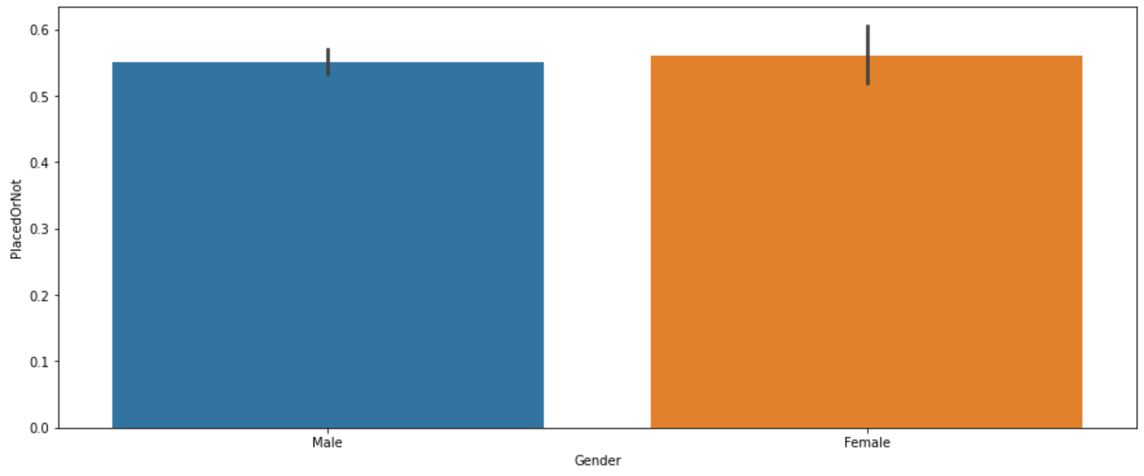
In [50]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df.Hostel, y = df.PlacedOrNot)
plt.show()
```



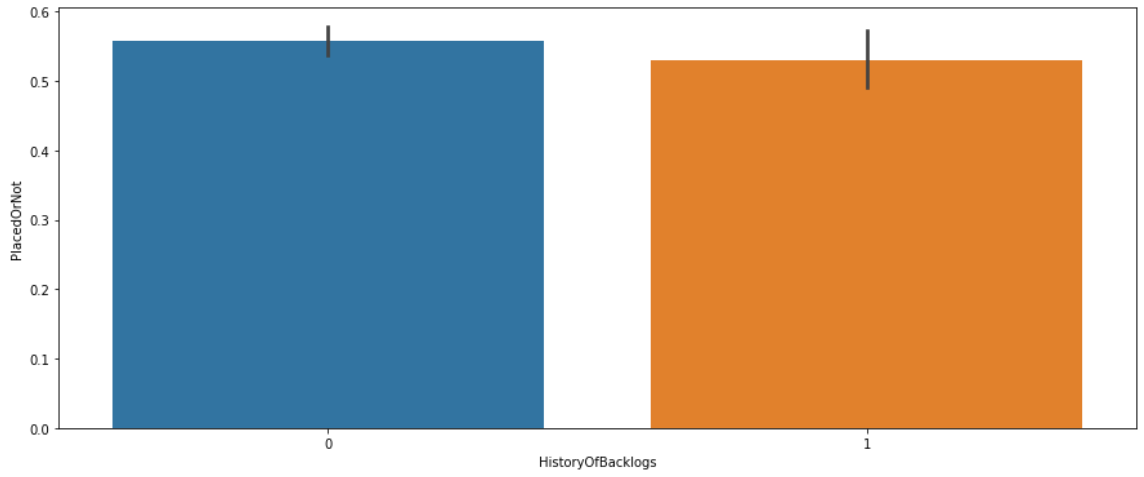
In [51]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df.Gender, y = df.PlacedOrNot)
plt.show()
```



In [52]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df.HistoryOfBacklogs, y = df.PlacedOrNot)
plt.show()
```



In [53]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [54]:

```
df.Gender = le.fit_transform(df.Gender)
df.Stream = le.fit_transform(df.Stream)
```

In [55]:

```
df.head()
```

Out[55]:

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	1	3	1	8	1	1	1
1	21	0	1	0	7	1	1	1
2	22	0	4	1	6	0	0	1
3	21	1	4	0	8	0	1	1
4	22	1	5	0	8	1	0	1

In [56]:

```
x = df.drop(['PlacedOrNot'], axis = 1)
```

In [57]:

```
y = df.PlacedOrNot
```

In [58]:

```
from sklearn import preprocessing
scaler = preprocessing.MinMaxScaler()
x = scaler.fit_transform(x)
```

In [59]:

```
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
```

In [60]:

```
from sklearn.model_selection import cross_val_score
```

In [61]:

```
cross_val_score(SVC(),x, y, cv = 3)
```

Out[61]:

```
array([0.80384226, 0.82305359, 0.90384615])
```

In [62]:

```
cross_val_score(DecisionTreeClassifier(), x, y, cv = 3)
```

Out[62]:

```
array([0.84428716, 0.84529828, 0.91497976])
```

In [63]:

```
cross_val_score(LogisticRegression(), x, y, cv = 3)
```

Out[63]:

```
array([0.71587462, 0.74418605, 0.83097166])
```

In [64]:

```
cross_val_score(RandomForestClassifier(n_estimators=50), x, y, cv = 3)
```

Out[64]:

```
array([0.84732053, 0.85237614, 0.8917004 ])
```

In [65]:

```
cross_val_score(KNeighborsClassifier(),x, y ,cv = 3)
```

Out[65]:

```
array([0.80788675, 0.80687563, 0.88461538])
```

In [66]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2)
```

In [67]:

```
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

Out[67]:

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [68]:

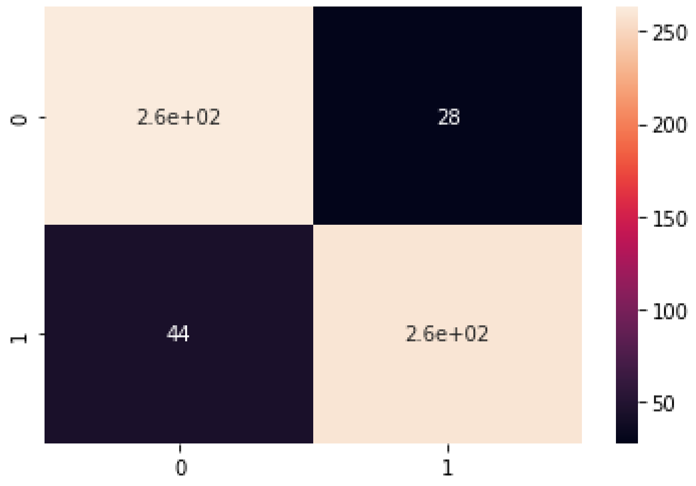
```
y_pred = model.predict(X_test)
```

In [69]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

In [70]:

```
sns.heatmap(cm, annot = True)
plt.show()
```



In [71]:

```
print("Training Accuracy :", model.score(X_train, y_train))
print("Testing Accuracy :", model.score(X_test, y_test))
```

```
Training Accuracy : 0.924114671163575
Testing Accuracy : 0.8787878787878788
```