📊 Career Change Prediction: EDA & Insights 🔍

Introduction

This dataset explores factors influencing career changes, including academic background, job satisfaction, industry growth, and more. With over 30,000 records and 22 attributes, the target variable, Likely to Change Occupation, predicts whether an individual is likely to switch careers.

This analysis aims to uncover patterns and provide actionable insights into career transitions.

Import Library ¶

```
In [3]: import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        import seaborn as sns
```

Uploading Csv fle

```
In [4]: df = pd.read_csv(r"C:\Users\Syed Arif\OneDrive\Desktop\Carrer Change\career_ch
        ange_prediction_dataset.csv")
```

Data Preprocessing

head()

head is used show to the By default = 5 rows in the dataset

In [5]:
```
df.head()
```

Out[5]:

	Field of Study	Current Occupation	Age	Gender	Years of Experience	Education Level	Industry Growth Rate	Job Satisfaction	Work-Life Balance
0	Medicine	Business Analyst	48	Male	7	High School	High	7	10
1	Education	Economist	44	Male	26	Master's	Low	10	3
2	Education	Biologist	21	Female	27	Master's	Low	8	3
3	Education	Business Analyst	33	Male	14	PhD	Medium	7	9
4	Arts	Doctor	28	Female	0	PhD	Low	3	1

5 rows × 23 columns

.tail()

tail is used to show rows by Descending order

In [6]:
```
df.tail()
```

Out[6]:

	Field of Study	Current Occupation	Age	Gender	Years of Experience	Education Level	Industry Growth Rate	Job Satisfaction	W Bala
38439	Biology	Business Analyst	24	Female	34	High School	Low	8	
38440	Mechanical Engineering	Artist	21	Female	24	High School	Low	2	
38441	Computer Science	Mechanical Engineer	35	Female	21	High School	High	4	
38442	Arts	Business Analyst	35	Male	11	PhD	Medium	9	
38443	Law	Mechanical Engineer	37	Male	23	Master's	Medium	6	

5 rows × 23 columns

shape

```
In [7]: df.shape
```

```
Out[7]: (38444, 23)
```

Columns

```
In [8]: df.columns
```

```
Out[8]: Index(['Field of Study', 'Current Occupation', 'Age', 'Gender',
               'Years of Experience', 'Education Level', 'Industry Growth Rate',
               'Job Satisfaction', 'Work-Life Balance', 'Job Opportunities', 'Salar
        y',
               'Job Security', 'Career Change Interest', 'Skills Gap',
               'Family Influence', 'Mentorship Available', 'Certifications',
               'Freelancing Experience', 'Geographic Mobility',
               'Professional Networks', 'Career Change Events', 'Technology Adoptio
        n',
               'Likely to Change Occupation'],
              dtype='object')
```

.dtypes

This Attribute show the data type of each column

```
In [9]: df.dtypes
```

```
Out[9]: Field of Study                  object
        Current Occupation              object
        Age                              int64
        Gender                          object
        Years of Experience              int64
        Education Level                 object
        Industry Growth Rate            object
        Job Satisfaction                 int64
        Work-Life Balance                int64
        Job Opportunities                int64
        Salary                           int64
        Job Security                     int64
        Career Change Interest           int64
        Skills Gap                       int64
        Family Influence                object
        Mentorship Available             int64
        Certifications                   int64
        Freelancing Experience           int64
        Geographic Mobility              int64
        Professional Networks            int64
        Career Change Events             int64
        Technology Adoption              int64
        Likely to Change Occupation      int64
        dtype: object
```

.unique()

In a column, It show the unique value of specific column.

```
In [10]: df["Current Occupation"].unique()
```

```
Out[10]: array(['Business Analyst', 'Economist', 'Biologist', 'Doctor', 'Lawyer',
                'Software Developer', 'Artist', 'Psychologist', 'Teacher',
                'Mechanical Engineer'], dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

.value_counts

It Shows all the unique values with their count

In [11]:

```
df["Current Occupation"].value_counts()
```

Out[11]:

```
Current Occupation
Software Developer     3892
Psychologist           3890
Doctor                 3888
Teacher                3886
Artist                 3881
Business Analyst       3858
Mechanical Engineer    3827
Lawyer                 3781
Biologist              3774
Economist              3767
Name: count, dtype: int64
```

isnull()

It shows the how many null values

In [12]:

```
df.isnull()
```

Out[12]:

	Field of Study	Current Occupation	Age	Gender	Years of Experience	Education Level	Industry Growth Rate	Job Satisfaction	Work-Life Balance
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
38439	False	False	False	False	False	False	False	False	False
38440	False	False	False	False	False	False	False	False	False
38441	False	False	False	False	False	False	False	False	False
38442	False	False	False	False	False	False	False	False	False
38443	False	False	False	False	False	False	False	False	False

38444 rows × 23 columns

.info()

To Show Data type of each column

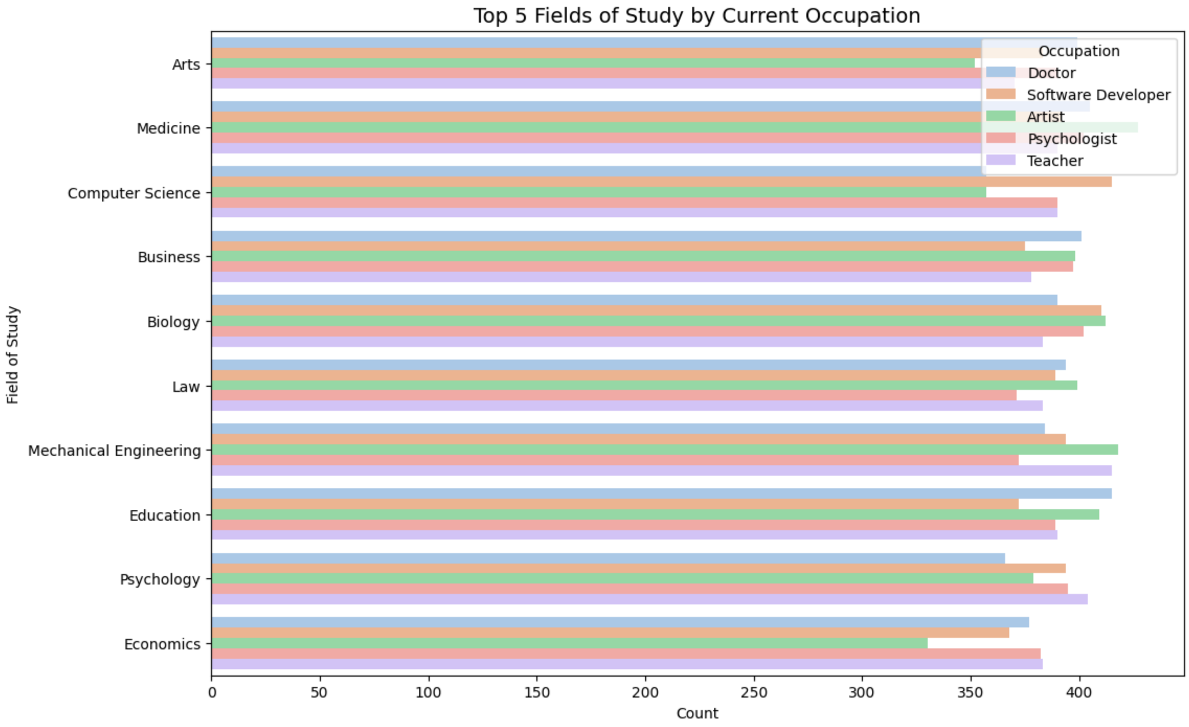
In [13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38444 entries, 0 to 38443
Data columns (total 23 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Field of Study               38444 non-null  object
 1   Current Occupation           38444 non-null  object
 2   Age                          38444 non-null  int64 
 3   Gender                       38444 non-null  object
 4   Years of Experience          38444 non-null  int64 
 5   Education Level              38444 non-null  object
 6   Industry Growth Rate         38444 non-null  object
 7   Job Satisfaction             38444 non-null  int64 
 8   Work-Life Balance            38444 non-null  int64 
 9   Job Opportunities            38444 non-null  int64 
 10  Salary                       38444 non-null  int64 
 11  Job Security                 38444 non-null  int64 
 12  Career Change Interest       38444 non-null  int64 
 13  Skills Gap                   38444 non-null  int64 
 14  Family Influence             28812 non-null  object
 15  Mentorship Available         38444 non-null  int64 
 16  Certifications               38444 non-null  int64 
 17  Freelancing Experience       38444 non-null  int64 
 18  Geographic Mobility          38444 non-null  int64 
 19  Professional Networks        38444 non-null  int64 
 20  Career Change Events         38444 non-null  int64 
 21  Technology Adoption          38444 non-null  int64 
 22  Likely to Change Occupation  38444 non-null  int64 
dtypes: int64(17), object(6)
memory usage: 6.7+ MB
```

1. Field of Study and Current Occupation (Top 5):

In [14]:
```python
top_occupations = df["Current Occupation"].value_counts().head(5).index
top_fields = df[df["Current Occupation"].isin(top_occupations)]

plt.figure(figsize=(12, 8))
sns.countplot(data=top_fields, y="Field of Study", hue="Current Occupation", palette="pastel")
plt.title("Top 5 Fields of Study by Current Occupation", fontsize=14)
plt.xlabel("Count")
plt.ylabel("Field of Study")
plt.legend(title="Occupation")
plt.show()
```

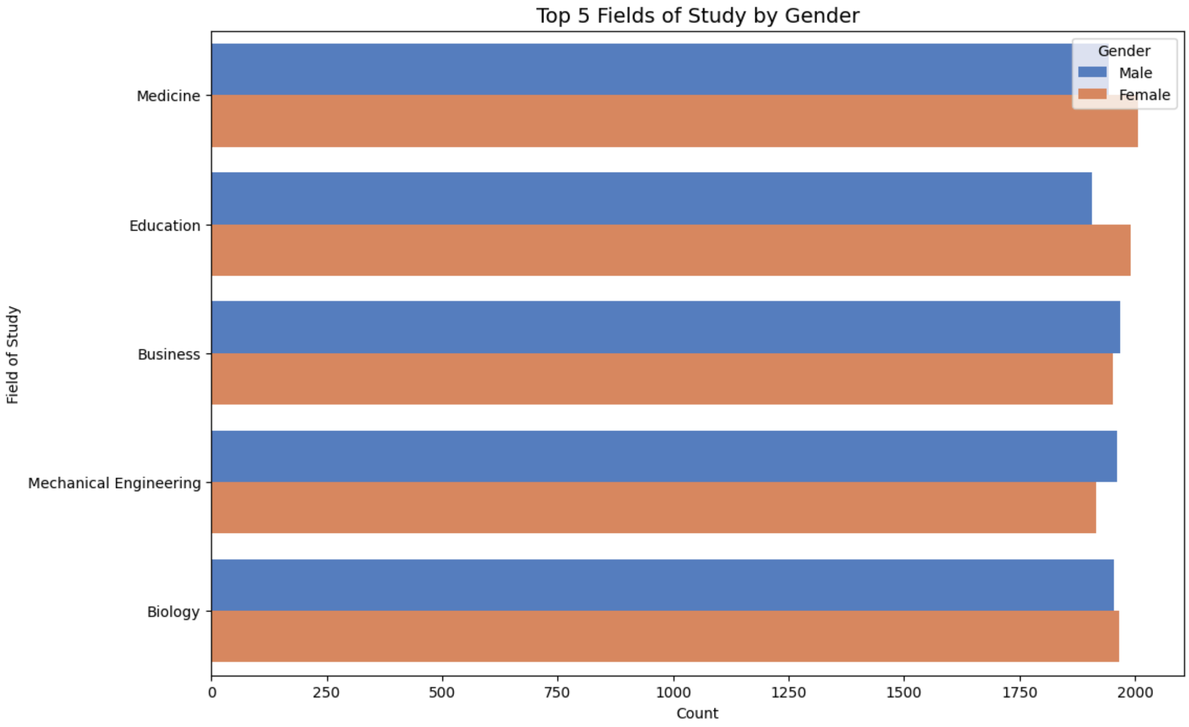


2. Field of Study and Gender Distribution (Top 5 Fields):

Question: Field of Study and Gender

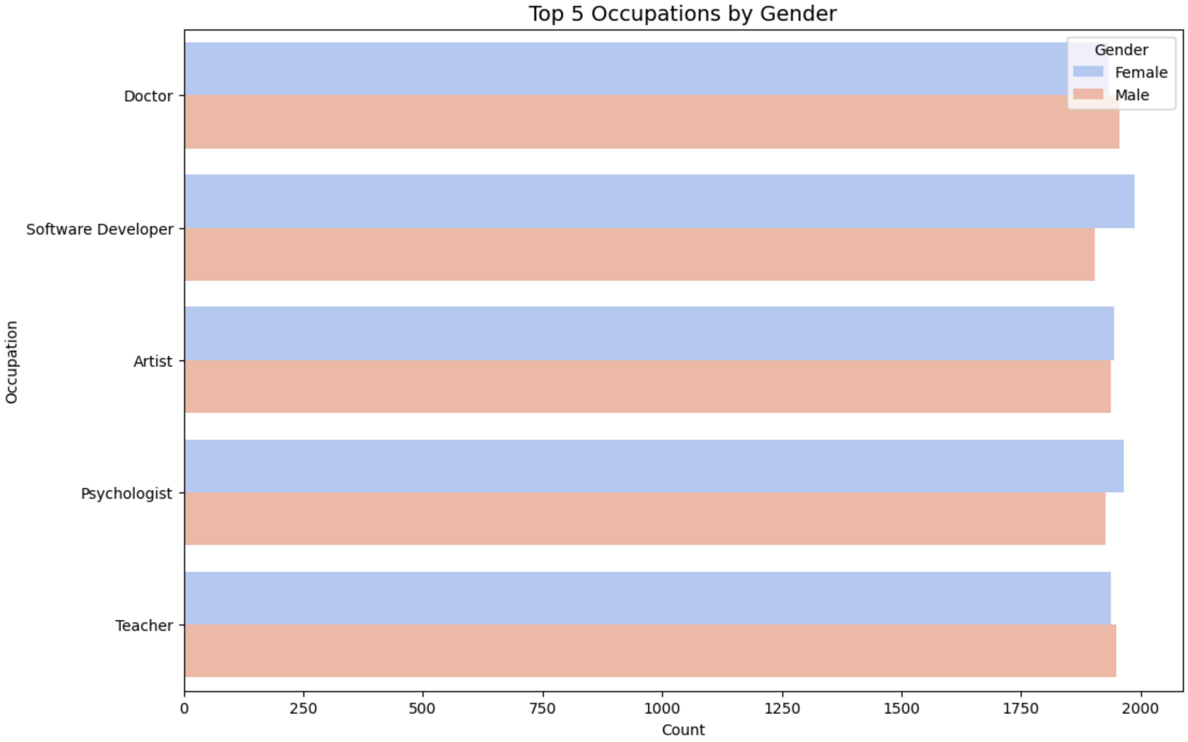
In [18]:

```
top_field_study = df["Field of Study"].value_counts().head(5).index
field_gender = df[df["Field of Study"].isin(top_field_study)]

plt.figure(figsize=(12, 8))
sns.countplot(data=field_gender, y="Field of Study", hue="Gender", palette="mu
ted")
plt.title("Top 5 Fields of Study by Gender", fontsize=14)
plt.xlabel("Count")
plt.ylabel("Field of Study")
plt.legend(title="Gender")
plt.show()
```



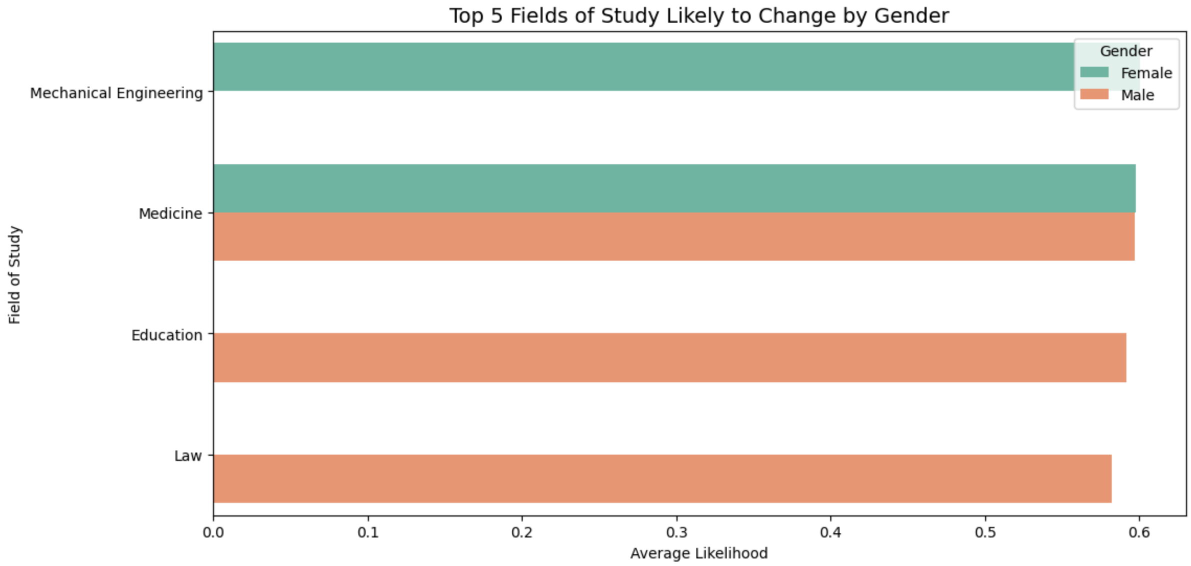
3. Occupation and Gender Distribution (Top 5 Occupations):

In [40]:
```
plt.figure(figsize=(12, 8))
sns.countplot(data=top_fields, y="Current Occupation", hue="Gender", palette
="coolwarm")
plt.title("Top 5 Occupations by Gender", fontsize=14)
plt.xlabel("Count")
plt.ylabel("Occupation")
plt.legend(title="Gender")
plt.show()
```



4. Top 5 Fields of Study Changed by Occupation and Gender:

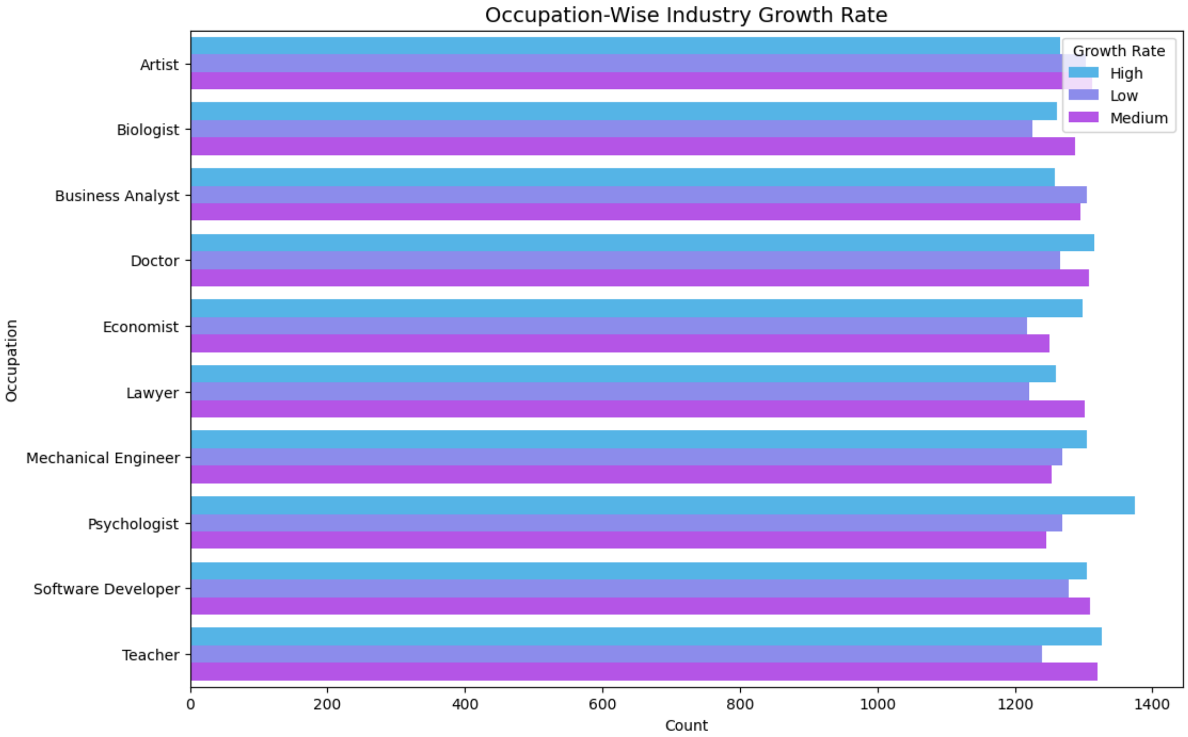
In [20]:
```python
study_change = df.groupby(["Field of Study", "Gender"])["Likely to Change Occupation"].mean().nlargest(5).reset_index()

plt.figure(figsize=(12, 6))
sns.barplot(data=study_change, x="Likely to Change Occupation", y="Field of Study", hue="Gender", palette="Set2")
plt.title("Top 5 Fields of Study Likely to Change by Gender", fontsize=14)
plt.xlabel("Average Likelihood")
plt.ylabel("Field of Study")
plt.legend(title="Gender")
plt.show()
```



5. Occupation-Wise Industry Growth Rate:

In [39]:

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Prepare the data for plotting
occupation_growth_rate = df.groupby(["Current Occupation", "Industry Growth Rate"]).size().reset_index(name="Count")

# Create a bar plot to show the count of each occupation by industry growth rate
plt.figure(figsize=(12, 8))
sns.barplot(data=occupation_growth_rate, x="Count", y="Current Occupation", hue="Industry Growth Rate", palette="cool")

# Customize the plot
plt.title("Occupation-Wise Industry Growth Rate", fontsize=14)
plt.xlabel("Count")
plt.ylabel("Occupation")
plt.legend(title="Growth Rate")
plt.show()
```



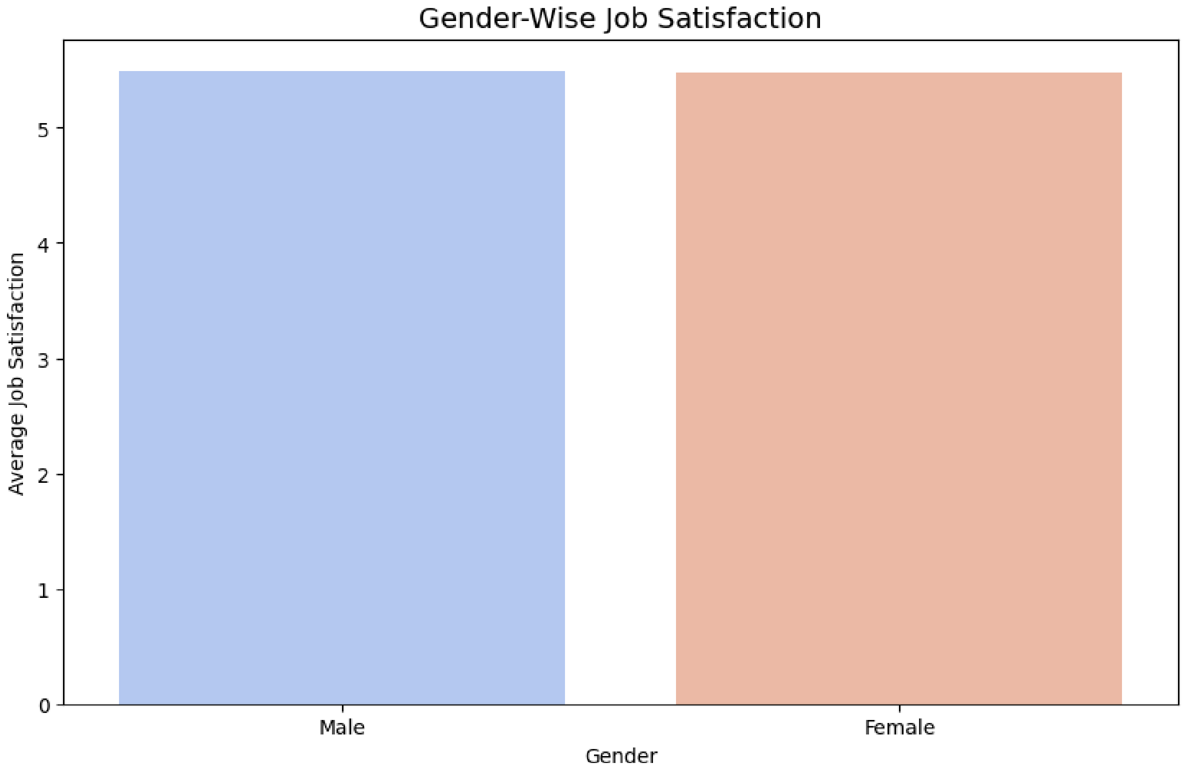
6. Gender-Wise Job Satisfaction:

Question: gender and calculate average job satisfaction?

In [38]:

```
import matplotlib.pyplot as plt
import seaborn as sns

# Create a bar plot to show the average job satisfaction by gender
plt.figure(figsize=(10, 6))
sns.barplot(x="Gender", y="Job Satisfaction", data=df, palette="coolwarm", ci=
None)

# Customize the plot
plt.title("Gender-Wise Job Satisfaction", fontsize=14)
plt.xlabel("Gender")
plt.ylabel("Average Job Satisfaction")
plt.show()
```

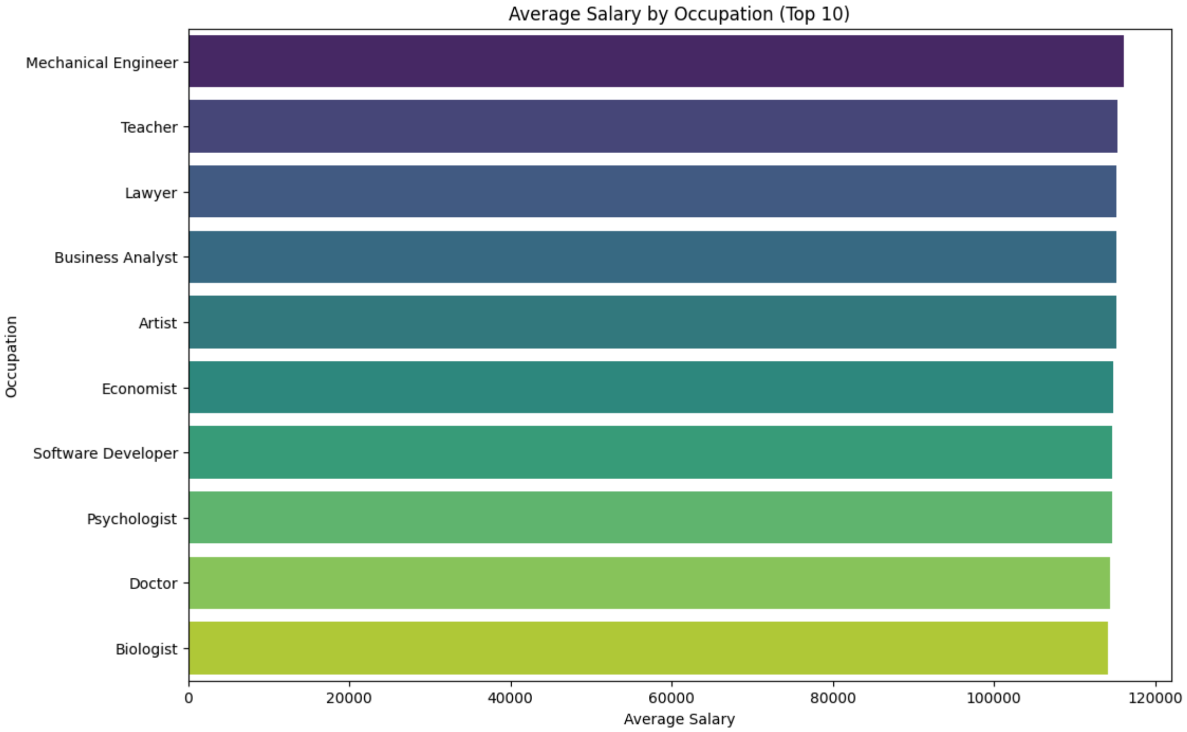
```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_7860\1016224037.py:6: FutureW
arning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.barplot(x="Gender", y="Job Satisfaction", data=df, palette="coolwarm",
ci=None)
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_7860\1016224037.py:6: FutureW
arning:

Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.

  sns.barplot(x="Gender", y="Job Satisfaction", data=df, palette="coolwarm",
ci=None)
```



7. Occupation-Wise Salary

In [27]:
```
occupation_salary_avg = df.groupby("Current Occupation")["Salary"].mean().sort_values(ascending=False).head(10)

plt.figure(figsize=(12, 8))
sns.barplot(
    x=occupation_salary_avg.values,
    y=occupation_salary_avg.index,
    palette="viridis",
)
plt.title("Average Salary by Occupation (Top 10)")
plt.xlabel("Average Salary")
plt.ylabel("Occupation")
plt.show()
```

```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_7860\2479832085.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(
```

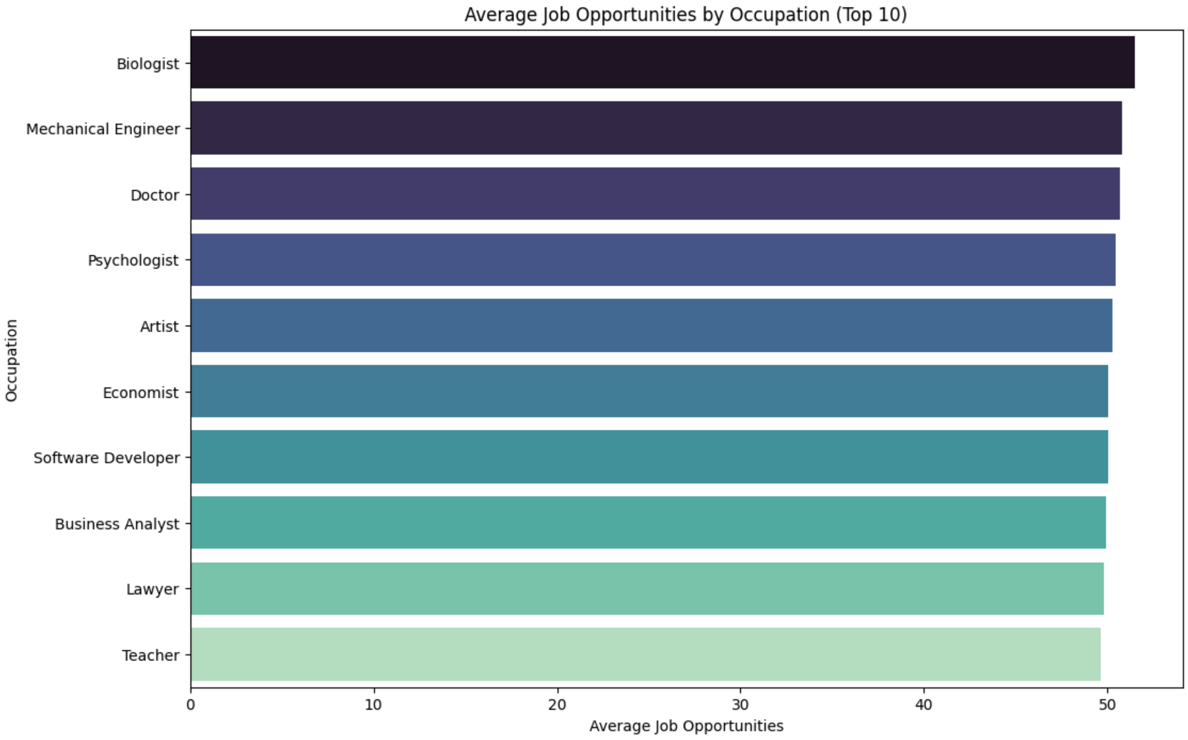


8. Occupation-Wise Job Opportunities:

In [29]:

```python
occupation_opportunities_avg = df.groupby("Current Occupation")["Job Opportuni
ties"].mean().sort_values(ascending=False).head(10)

plt.figure(figsize=(12, 8))
sns.barplot(
    x=occupation_opportunities_avg.values,
    y=occupation_opportunities_avg.index,
    palette="mako",
)
plt.title("Average Job Opportunities by Occupation (Top 10)")
plt.xlabel("Average Job Opportunities")
plt.ylabel("Occupation")
plt.show()
```

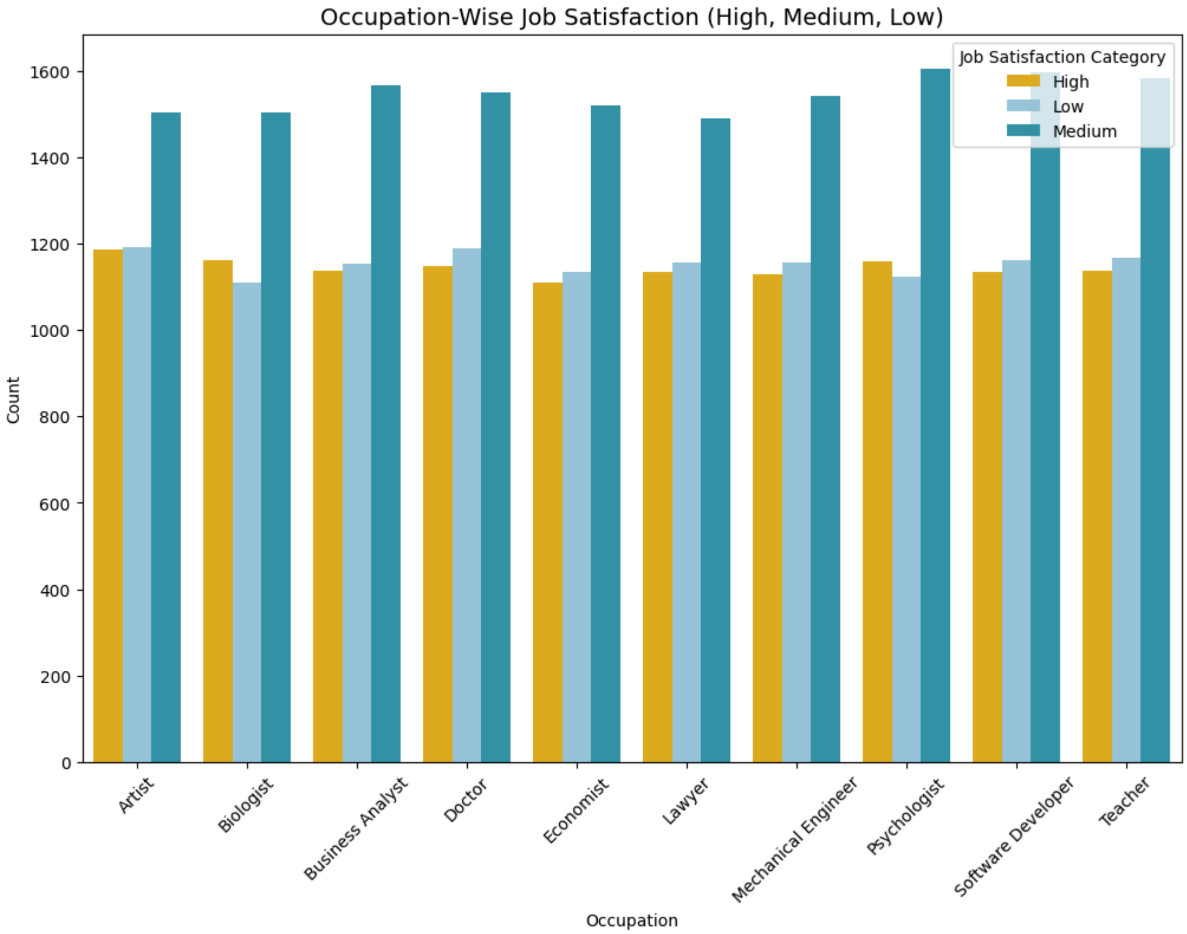
```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_7860\582844519.py:4: FutureWa
rning:

Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the sa
me effect.

  sns.barplot(
```



9. Occupation-Wise Job Satisfaction:

In [36]:

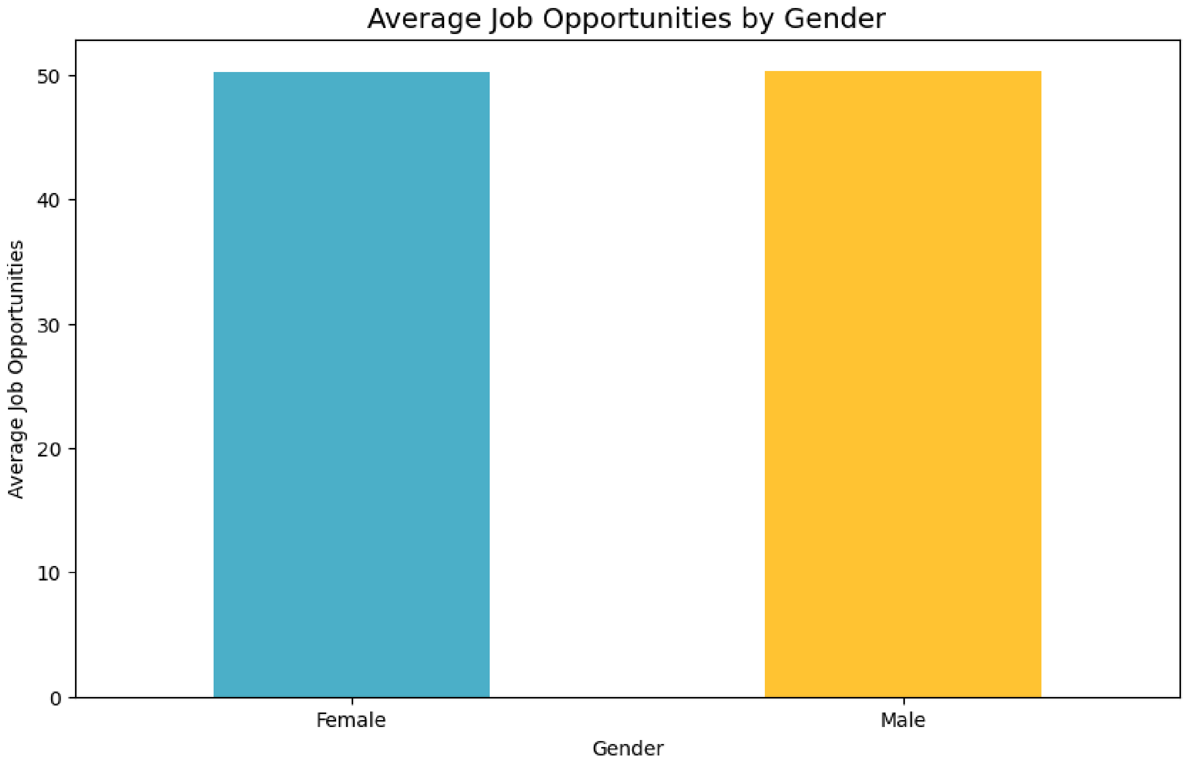
```
# Define a function to map numeric scores to satisfaction levels
def map_satisfaction(value):
    if value <= 3:
        return "Low"
    elif 4 <= value <= 7:
        return "Medium"
    else:
        return "High"

# Filter occupations ranked from 1 to 10 (e.g., top 10 by frequency)
top_occupations = df["Current Occupation"].value_counts().head(10).index
filtered_data = df[df["Current Occupation"].isin(top_occupations)]

# Map numeric satisfaction levels to categories
filtered_data["Job Satisfaction Category"] = filtered_data["Job Satisfaction"].apply(map_satisfaction)

# Prepare the data for visualization
occupation_satisfaction_counts = (
    filtered_data.groupby(["Current Occupation", "Job Satisfaction Category"])
    .size()
    .reset_index(name="Count")
)

# Plot a stacked bar chart using hue (similar to the first code example)
plt.figure(figsize=(12, 8))
sns.barplot(
    data=occupation_satisfaction_counts,
    x="Current Occupation",
    y="Count",
    hue="Job Satisfaction Category",
    palette=["#ffb703", "#8ecae6", "#219ebc"]
)

# Customize the plot
plt.title("Occupation-Wise Job Satisfaction (High, Medium, Low)", fontsize=14)
plt.xlabel("Occupation")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.legend(title="Job Satisfaction Category")
plt.show()
```



10. Occupation-Wise Industry Growth Rate

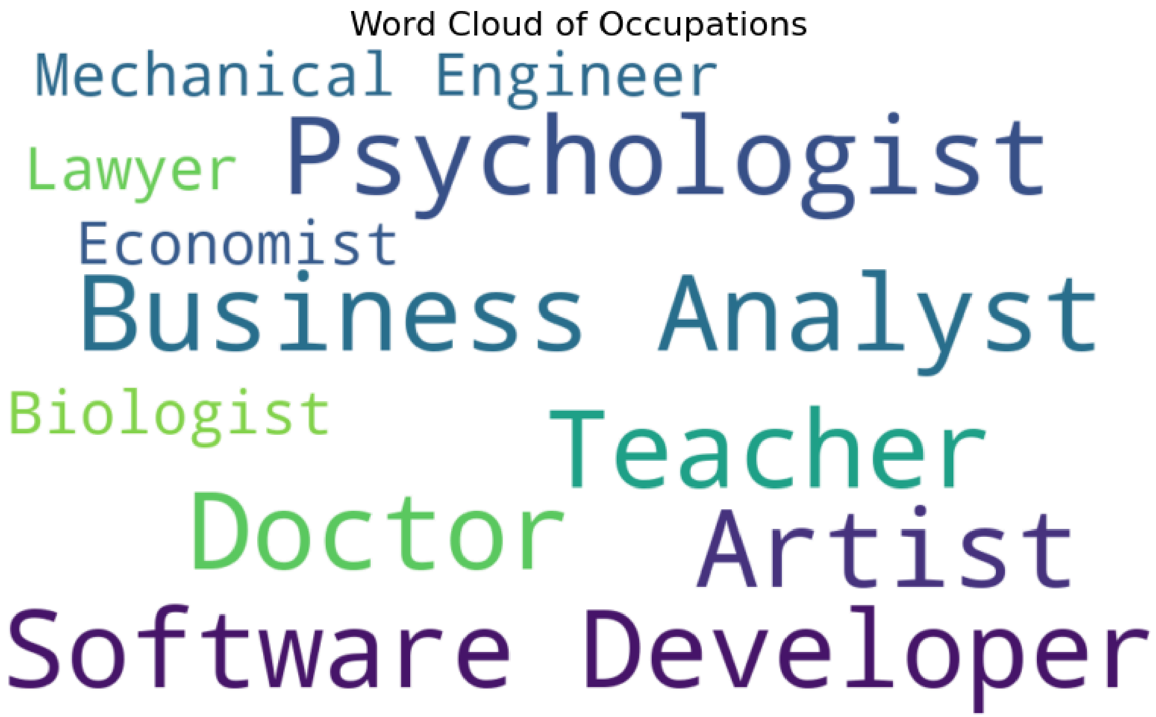
Question: How does job satisfaction vary across occupations and genders?

In [32]:
```
gender_opportunities_mean = df.groupby("Gender")["Job Opportunities"].mean()

plt.figure(figsize=(10, 6))
gender_opportunities_mean.plot(
    kind="bar",
    color=["#219ebc", "#ffb703"],
    alpha=0.8,
)
plt.title("Average Job Opportunities by Gender", fontsize=14)
plt.xlabel("Gender")
plt.ylabel("Average Job Opportunities")
plt.xticks(rotation=0)
plt.show()
```



In [51]:

```python
# Import necessary libraries
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Generate a word cloud based on the frequency of 'Current Occupation'
wordcloud = WordCloud(width=1000, height=600, background_color='white', colormap='viridis').generate_from_frequencies(df['Current Occupation'].value_counts())

# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud of Occupations', fontsize=16)
plt.axis('off')  # Disable the axis for better visualization
plt.show()
```


Word Cloud of Occupations
Mechanical Engineer
Lawyer
Psychologist
Economist
Business Analyst
Biologist
Teacher
Doctor
Artist
Software Developer