Credit Risk Analysis using Python

Project Overview

This project focuses on analyzing credit risk data to identify patterns in loan defaults,customer risk levels , and important factors affecting creditworthiness.

```
Using Python libraries like **Pandas ,NumPy, Seaborn and
matplotlib** , we:
```

--Cleaned and prepared the data

--Engineered new useful features(like loan-to income ratio)

--Visualized key relationships using charts

--Segments customers into **Low** ,**Medium** , and **High Risk** categories

This analysis helps financial institutions understand customer profiles and improve loan decision-making.

Dataset Descriptions:

The dataset includes the following key columns:

--'person_age','person_income',"person_home_ownership",'person_emp_length'

--'loan_intent','loan_grade' , 'loan_amnt','loan_int_rate'

--'loan_status','loan_percent_income'

--'cb_income_default_on_file','cb_person_cred_hist_length'

Project Goals:

--identify factors contributing to loan default

--Segment customers by risk

--Support decision-making with clean visuals and insights

Tools Used:

python:pandas , numpy,seaborn , matplotlib

Advanced Excel(for pivot tables and charts)

Power BI(optional dashboard)

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

In [5]:
```
df=pd.read_csv('credit_risk_dataset.csv')
df.head()
```

Out[5]:

	person_age	person_income	person_home_ownership	person_emp_length	loan_intent	l
0	22	59000	RENT	123.0	PERSONAL	
1	21	9600	OWN	5.0	EDUCATION	
2	25	9600	MORTGAGE	1.0	MEDICAL	
3	23	65500	RENT	4.0	MEDICAL	
4	24	54400	RENT	8.0	MEDICAL	

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32581 entries, 0 to 32580
Data columns (total 12 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   person_age                  32581 non-null  int64
 1   person_income               32581 non-null  int64
 2   person_home_ownership       32581 non-null  object
 3   person_emp_length           31686 non-null  float64
 4   loan_intent                 32581 non-null  object
 5   loan_grade                  32581 non-null  object
 6   loan_amnt                   32581 non-null  int64
 7   loan_int_rate               29465 non-null  float64
 8   loan_status                 32581 non-null  int64
 9   loan_percent_income         32581 non-null  float64
 10  cb_person_default_on_file   32581 non-null  object
 11  cb_person_cred_hist_length  32581 non-null  int64
dtypes: float64(3), int64(5), object(4)
memory usage: 3.0+ MB
```

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
person_age                      0
person_income                   0
person_home_ownership           0
person_emp_length             895
loan_intent                     0
loan_grade                      0
loan_amnt                       0
loan_int_rate                3116
loan_status                     0
loan_percent_income             0
cb_person_default_on_file       0
cb_person_cred_hist_length      0
dtype: int64
```

In [8]:
```
df['person_income']=df['person_income'].fillna(df['person_income'].mean())
df['person_age'] = df['person_age'].fillna(df['person_age'].mean())
df['loan_percent_income'] = df['loan_percent_income'].fillna(df['loan_percent_incom
df['cb_person_cred_hist_length'] = df['cb_person_cred_hist_length'].fillna(df['cb_p
```

In [9]:
```
df['person_home_ownership'] = df['person_home_ownership'].astype(str).str.lower().s
df['loan_intent'] = df['loan_intent'].astype(str).str.lower().str.strip()
df['loan_grade'] = df['loan_grade'].astype(str).str.lower().str.strip()
df['loan_status'] = df['loan_status'].astype(str).str.lower().str.strip()
df['cb_person_default_on_file'] = df['cb_person_default_on_file'].astype(str).str.l
df['person_emp_length'] = df['person_emp_length'].astype(str).str.lower().str.strip
```

In [10]:
```
df.isnull().sum()
df.info()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32581 entries, 0 to 32580
Data columns (total 12 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   person_age                  32581 non-null  int64
 1   person_income               32581 non-null  int64
 2   person_home_ownership       32581 non-null  object
 3   person_emp_length           32581 non-null  object
 4   loan_intent                 32581 non-null  object
 5   loan_grade                  32581 non-null  object
 6   loan_amnt                   32581 non-null  int64
 7   loan_int_rate               29465 non-null  float64
 8   loan_status                 32581 non-null  object
 9   loan_percent_income         32581 non-null  float64
 10  cb_person_default_on_file   32581 non-null  object
 11  cb_person_cred_hist_length  32581 non-null  int64
dtypes: float64(2), int64(4), object(6)
memory usage: 3.0+ MB
```

Out[10]:

	person_age	person_income	person_home_ownership	person_emp_length	loan_intent	l
0	22	59000	rent	123.0	personal	
1	21	9600	own	5.0	education	
2	25	9600	mortgage	1.0	medical	
3	23	65500	rent	4.0	medical	
4	24	54400	rent	8.0	medical	

In [11]:
```
df['loan_to_income_ratio'] = df['loan_amnt']
```

In [12]:
```
def clean_emp_length(val):
    if pd.isnull(val):
        return 0
    elif '<' in val:
        return 0.5
    elif"10+" in val:
        return 10
    else:
        try:
            return float(val.split()[0])
        except:
            return 0
df['emp_length_cleaned'] = df['person_emp_length'].apply(clean_emp_length)
```

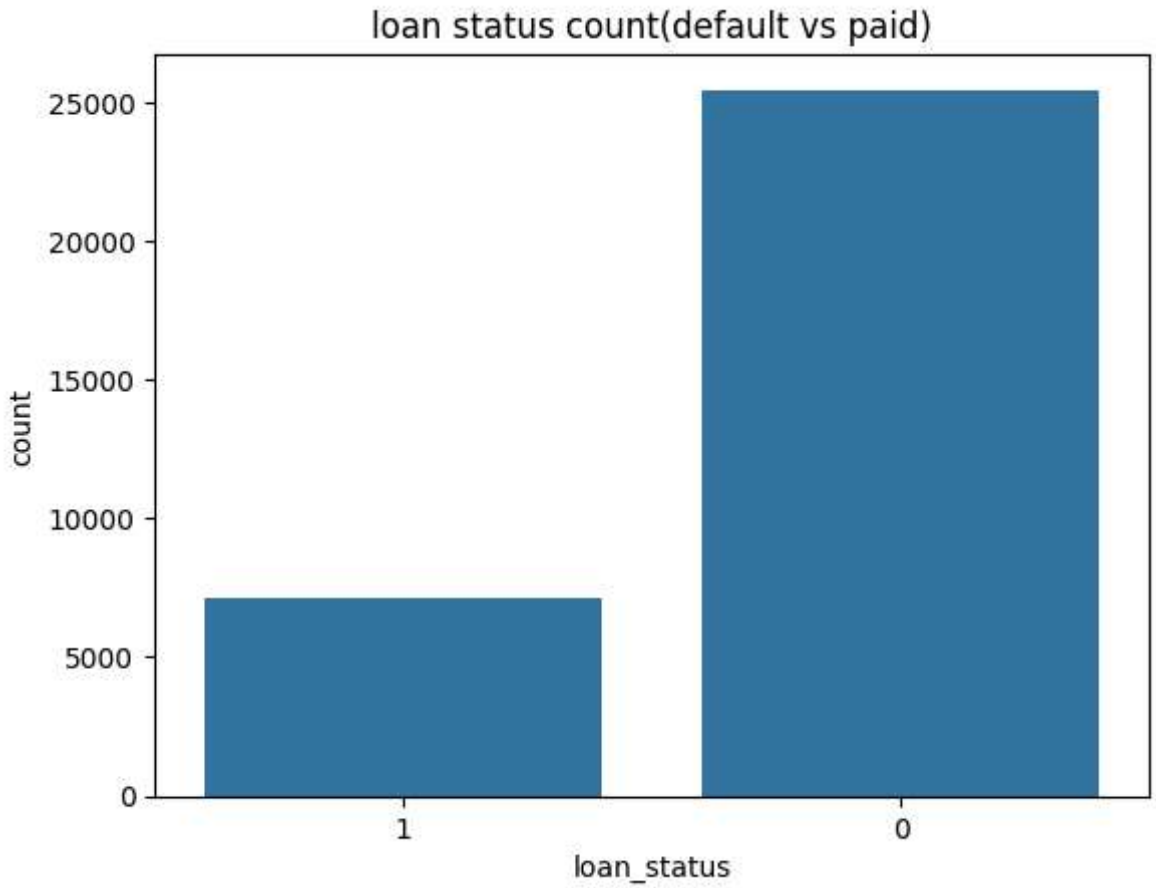
In [13]:
```
df['default_flag'] = df['loan_status'].apply(lambda x:1 if x == 'default' else 0)
```

In [14]:
```
df[['loan_amnt','person_income','loan_to_income_ratio','emp_length_cleaned' , 'defa
```

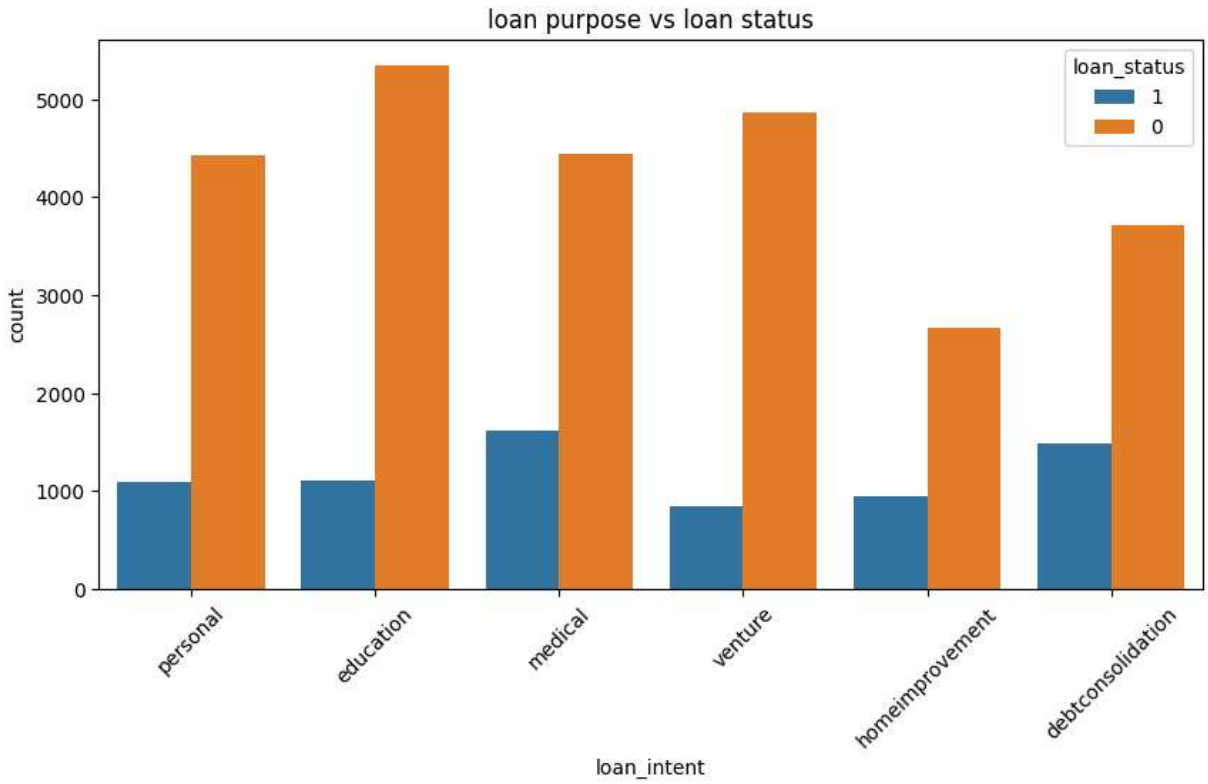
Out[14]:

	loan_amnt	person_income	loan_to_income_ratio	emp_length_cleaned	default_flag
0	35000	59000	35000	123.0	0
1	1000	9600	1000	5.0	0
2	5500	9600	5500	1.0	0
3	35000	65500	35000	4.0	0
4	35000	54400	35000	8.0	0

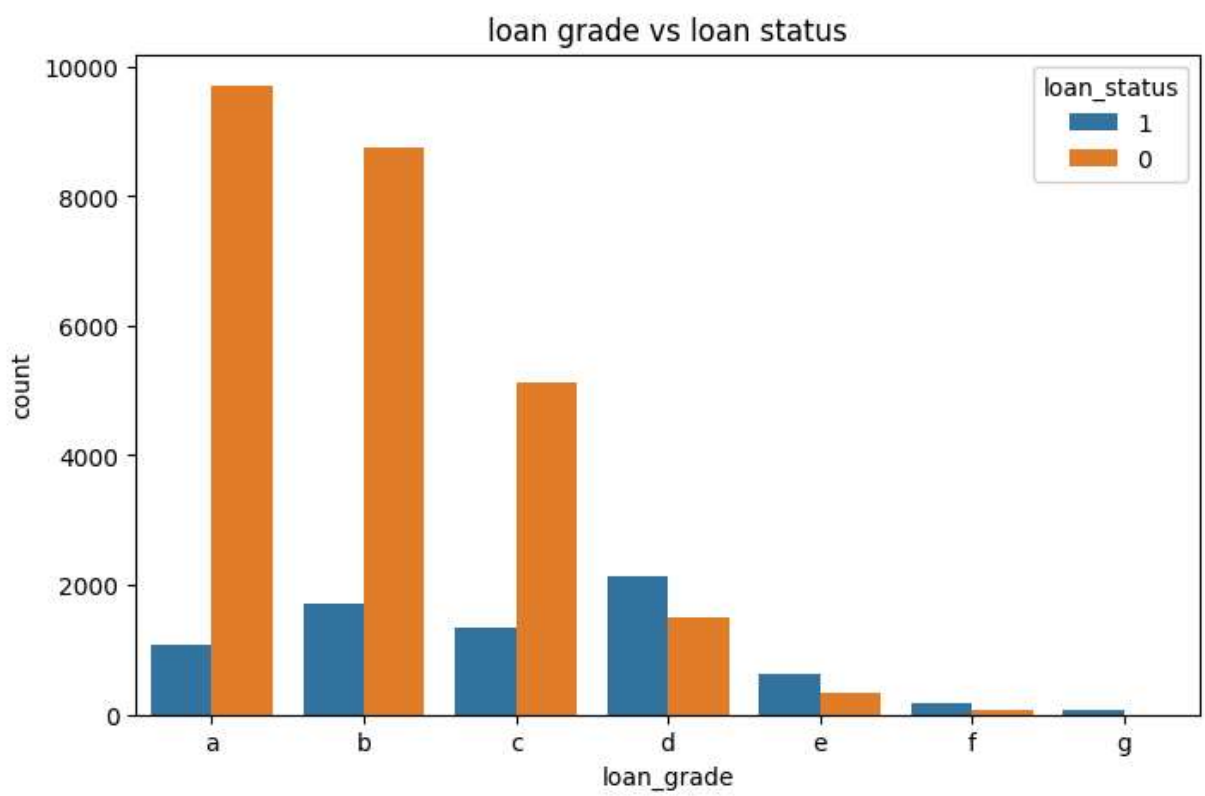
In [15]:
```
sns.countplot(x='loan_status',data=df)
plt.title('loan status count(default vs paid)')
plt.show()
```



In [16]:
```
plt.figure(figsize =(10,5))
sns.countplot(x='loan_intent',hue='loan_status',data=df)
plt.title('loan purpose vs loan status')
plt.xticks(rotation=45)
plt.show()
```

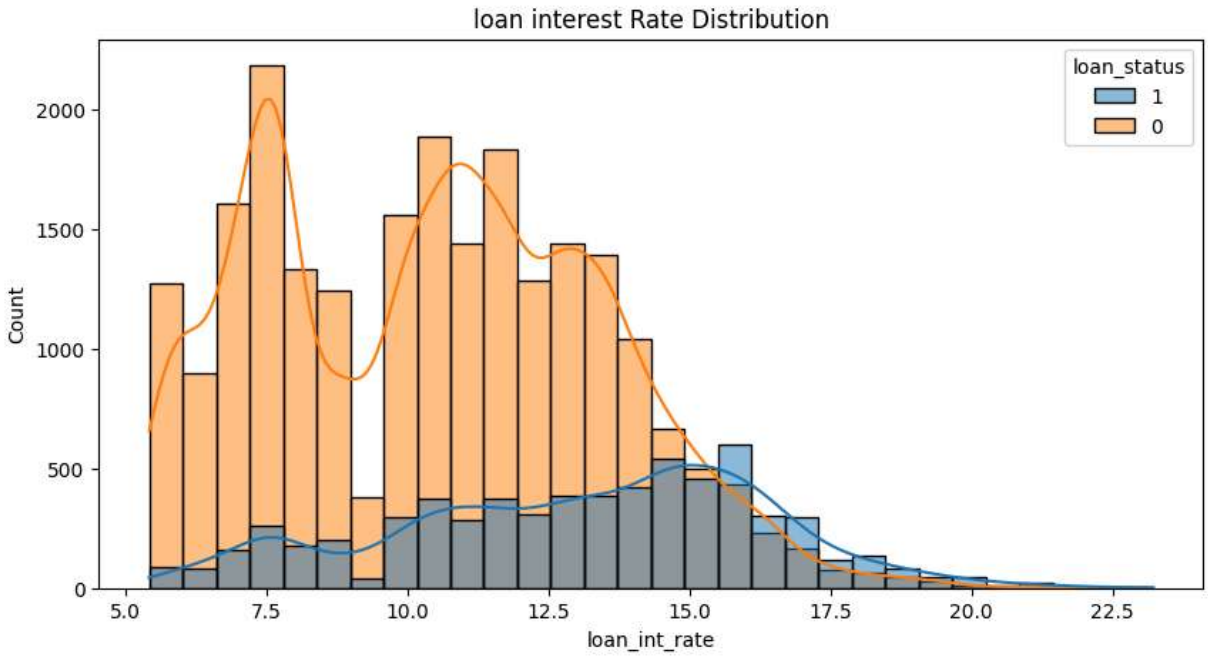


In [17]:
```
plt.figure(figsize=(8,5))
sns.countplot(x='loan_grade' , hue='loan_status',data =df , order = sorted(df['loan
plt.title('loan grade vs loan status')
plt.show()
```

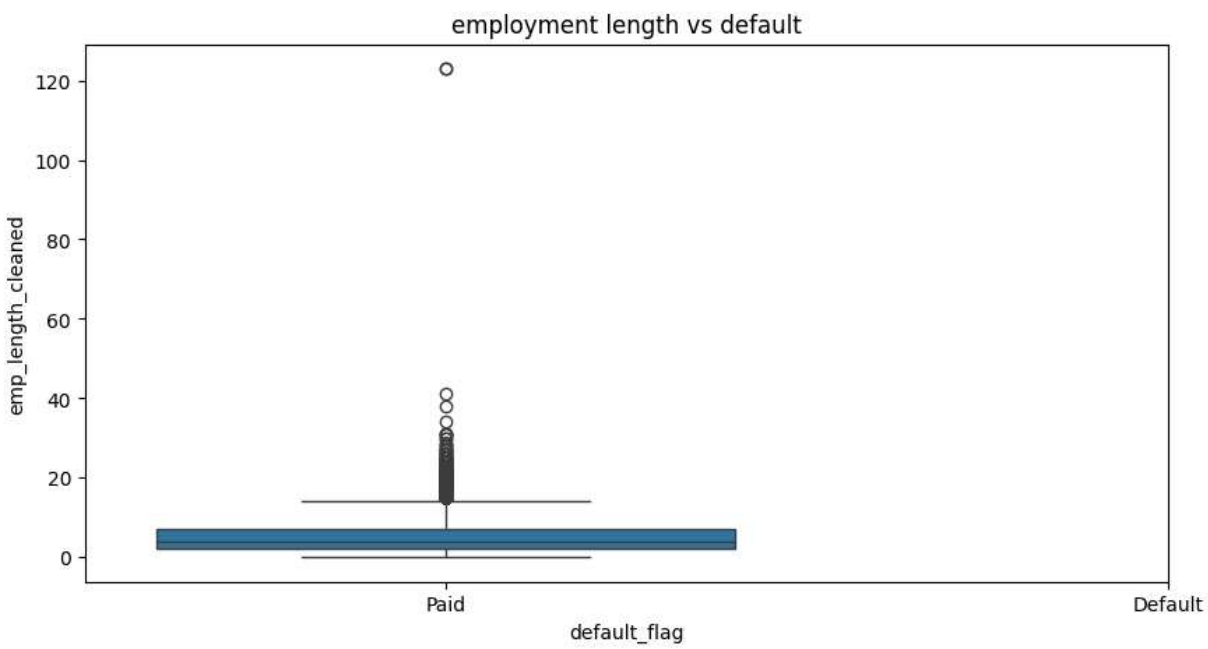


In [18]:
```
plt.figure(figsize=(10,5))
sns.histplot(data=df,x='loan_int_rate' ,hue='loan_status',bins=30,kde=True)
```

```
plt.title('loan interest Rate Distribution')
plt.show()
```



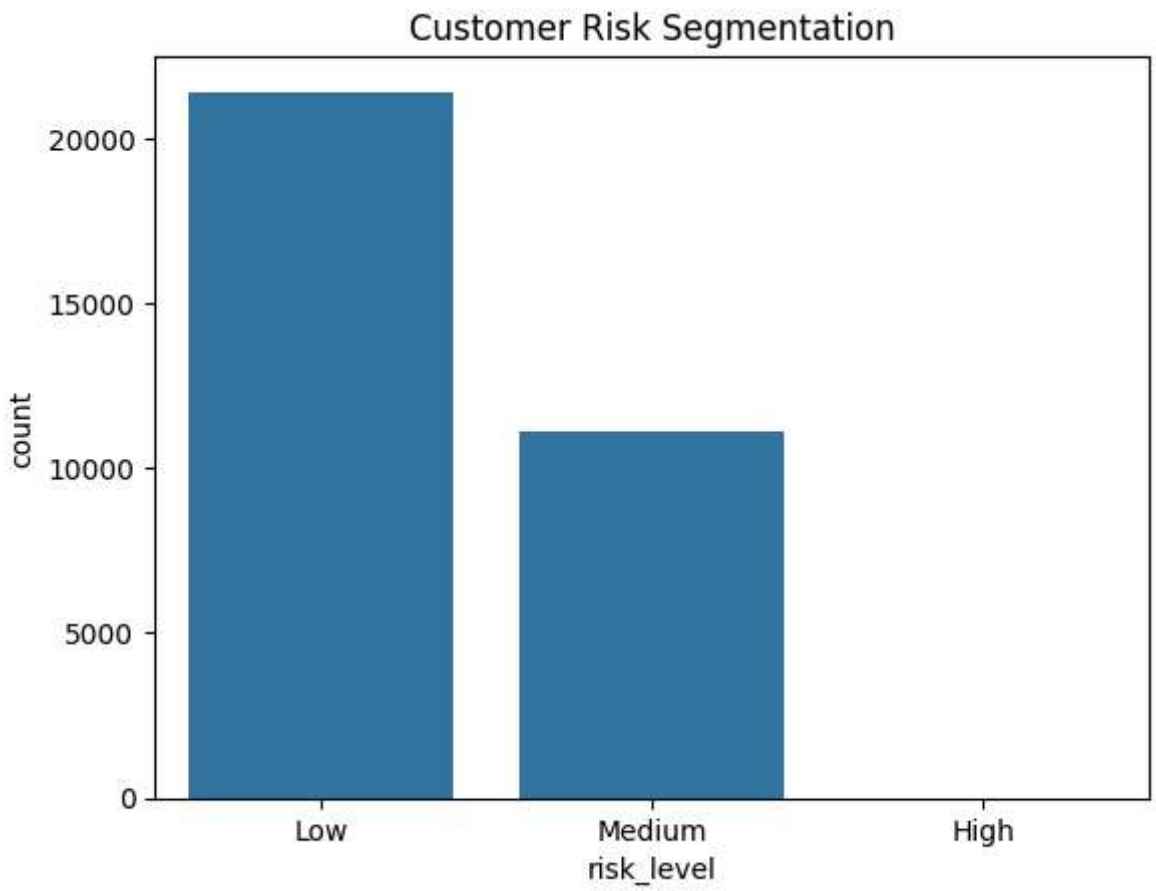
In [19]:
```
plt.figure(figsize =(10,5))
sns.boxplot(x='default_flag',y='emp_length_cleaned',data=df)
plt.title('employment length vs default')
plt.xticks([0,1],['Paid','Default'])
plt.show()
```



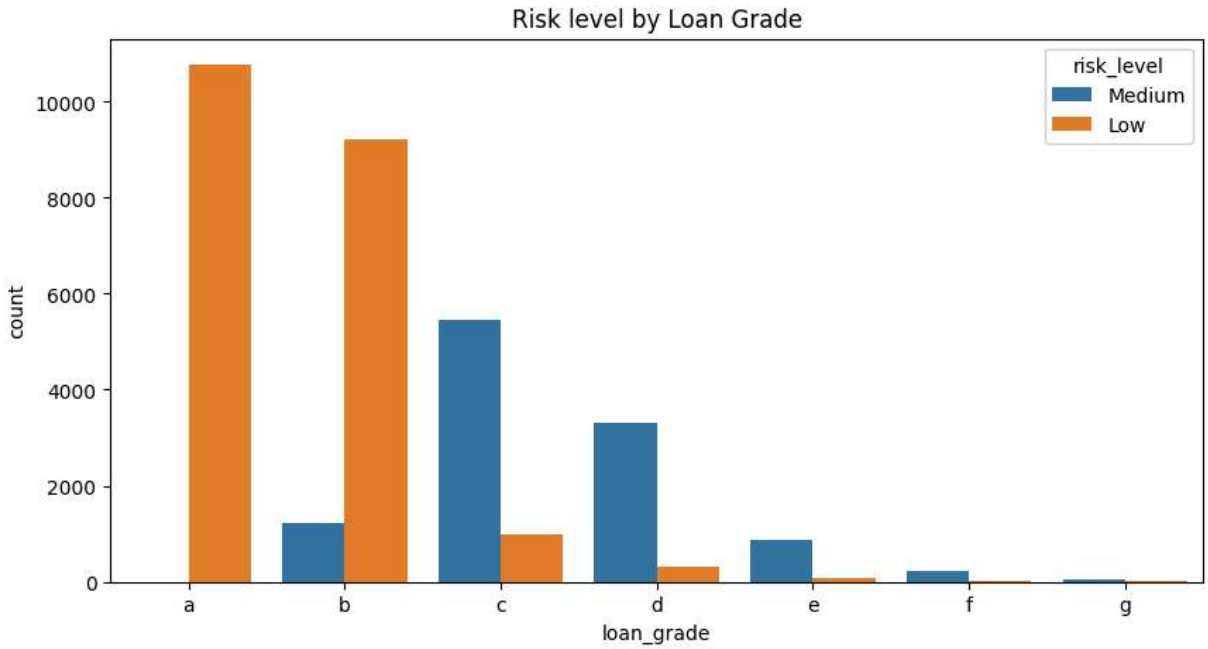
In [20]:
```
def get_risk_level(row):
    if row['loan_to_income_ratio']>0.4 and row['loan_int_rate']>15 and row['default
        return 'High'
    elif row['loan_to_income_ratio'] >0.3 and row['loan_int_rate']>12:
        return'Medium'
    else:
```

```
        return'Low'
df['risk_level'] = df.apply(get_risk_level , axis = 1)
```

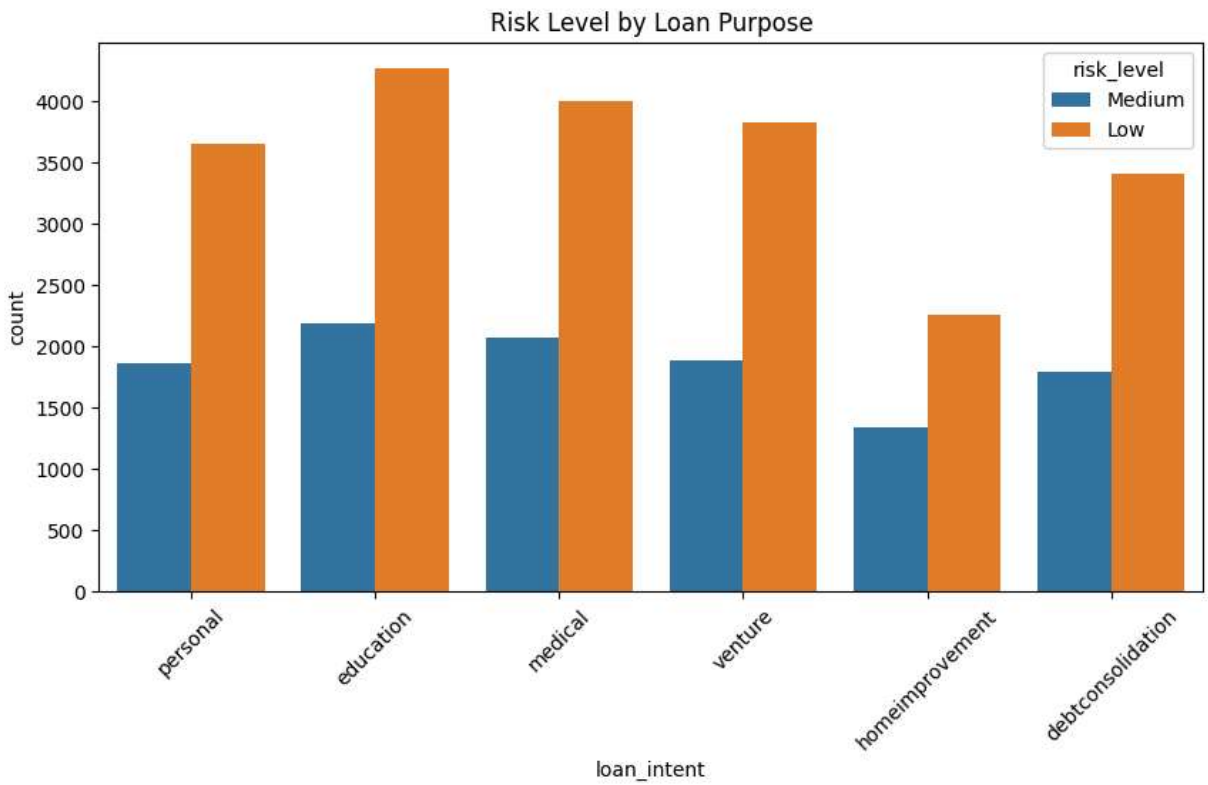
In [21]:
```
sns.countplot(x='risk_level',data = df ,order=['Low','Medium','High'])
plt.title("Customer Risk Segmentation")
plt.show()
```



In [22]:
```
plt.figure(figsize=(10,5))
sns.countplot(x='loan_grade',hue='risk_level',data=df,order=sorted(df['loan_grade']
plt.title("Risk level by Loan Grade")
plt.show()
```



In [23]:
```
plt.figure(figsize=(10,5))
sns.countplot(x='loan_intent', hue='risk_level',data=df)
plt.title('Risk Level by Loan Purpose')
plt.xticks(rotation=45)
plt.show()
```



In [24]:
```
df.to_excel("cleaned_credit_data.xlsx", index=False)
```

In [25]:

Out[25]:

	loan_amnt	person_income	loan_to_income_ratio	emp_length_cleaned	default_flag
0	35000	59000	35000	123.0	0
1	1000	9600	1000	5.0	0
2	5500	9600	5500	1.0	0
3	35000	65500	35000	4.0	0
4	35000	54400	35000	8.0	0

In [ ]: