In [15]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

In [16]:
```
# Load your datasets
fact_loan = pd.read_csv("C:/Users/admin/Desktop/projects/Loan Performance & Customer Segmentation/fact_loans.csv
dim_customer = pd.read_csv("C:/Users/admin/Desktop/projects/Loan Performance & Customer Segmentation/dim_customer
dim_branch = pd.read_csv("C:/Users/admin/Desktop/projects/Loan Performance & Customer Segmentation/dim_branch.cs
```

Inspect Structure

In [17]:
```
# Basic info
print(fact_loan.info())
print(dim_customer.info())
print(dim_branch.info())

# Check top rows
print(fact_loan.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   LoanID              1000 non-null   int64  
 1   CustomerID          1000 non-null   int64  
 2   BranchID            1000 non-null   int64  
 3   LoanAmount          1000 non-null   float64
 4   InterestRate        1000 non-null   float64
 5   Term                1000 non-null   int64  
 6   StartDate           1000 non-null   object 
 7   EndDate             1000 non-null   object 
 8   Status              1000 non-null   object 
 9   AmountPaid          1000 non-null   float64
 10  OutstandingBalance  1000 non-null   float64
dtypes: float64(4), int64(4), object(3)
memory usage: 86.1+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 6 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   CustomerID        500 non-null    int64 
 1   Name              500 non-null    object
 2   Age               500 non-null    int64 
 3   Gender            500 non-null    object
 4   Income            500 non-null    int64 
 5   EmploymentStatus  500 non-null    object
dtypes: int64(3), object(3)
memory usage: 23.6+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   BranchID    10 non-null     int64 
 1   BranchName  10 non-null     object
 2   City        10 non-null     object
 3   Region      10 non-null     object
dtypes: int64(1), object(3)
memory usage: 452.0+ bytes
None
   LoanID  CustomerID  BranchID  LoanAmount  InterestRate  Term   StartDate  \
0       1         364         2    29749.87          6.48    36  2022-08-07   
1       2         475         3    40716.34          9.11    12  2024-02-25   
2       3         122         8     6182.78          9.65    36  2024-12-16   
3       4         283         3    32537.68          4.03    36  2023-03-22   
4       5         368         8    28656.30         12.23    60  2025-02-02   

      EndDate       Status  AmountPaid  OutstandingBalance  
0  2025-08-07         Paid    25865.94                0.00  
1  2025-02-25    Defaulted    38014.91             2701.43  
2  2027-12-16  In Progress     1236.56             4946.22  
3  2026-03-22  In Progress     6507.54            26030.14  
4  2030-02-02     Approved     5731.26            22925.04  
```

Check Missing Values

In [18]:
```
# Nulls check
print(fact_loan.isnull().sum())
print(dim_customer.isnull().sum())
print(dim_branch.isnull().sum())
```

```
LoanID               0
CustomerID           0
BranchID             0
LoanAmount           0
InterestRate         0
Term                 0
StartDate            0
EndDate              0
Status               0
AmountPaid           0
OutstandingBalance   0
dtype: int64
CustomerID          0
Name                0
Age                 0
Gender              0
Income              0
EmploymentStatus    0
dtype: int64
BranchID      0
BranchName    0
City          0
Region        0
dtype: int64
```

Handle Duplicates

In [19]:
```
# Remove exact duplicates
fact_loan.drop_duplicates(inplace=True)
dim_customer.drop_duplicates(subset='CustomerID', inplace=True)
```

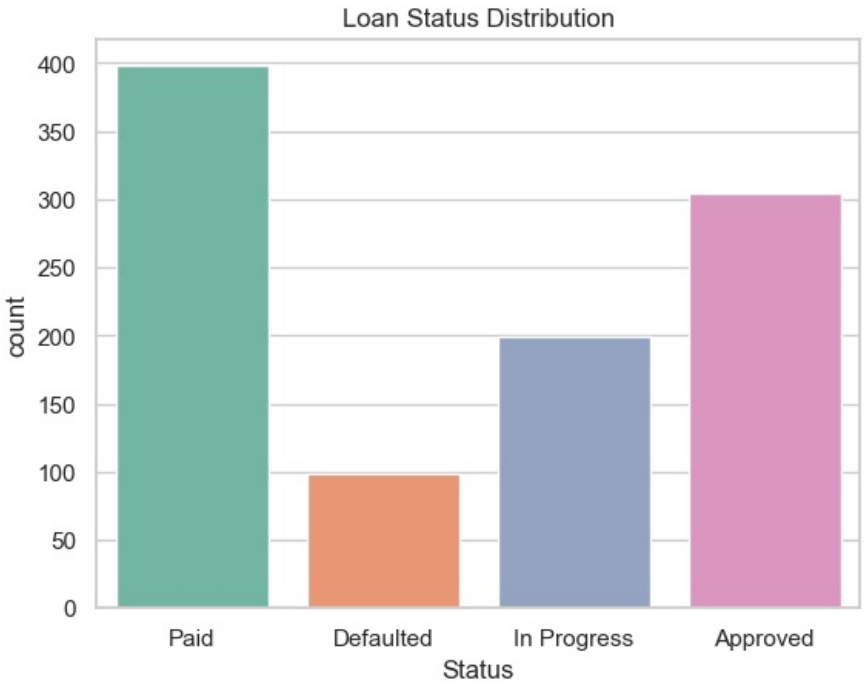
EXPLORATORY DATA ANALYSIS (EDA)

In [20]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Setup
sns.set(style="whitegrid")
```

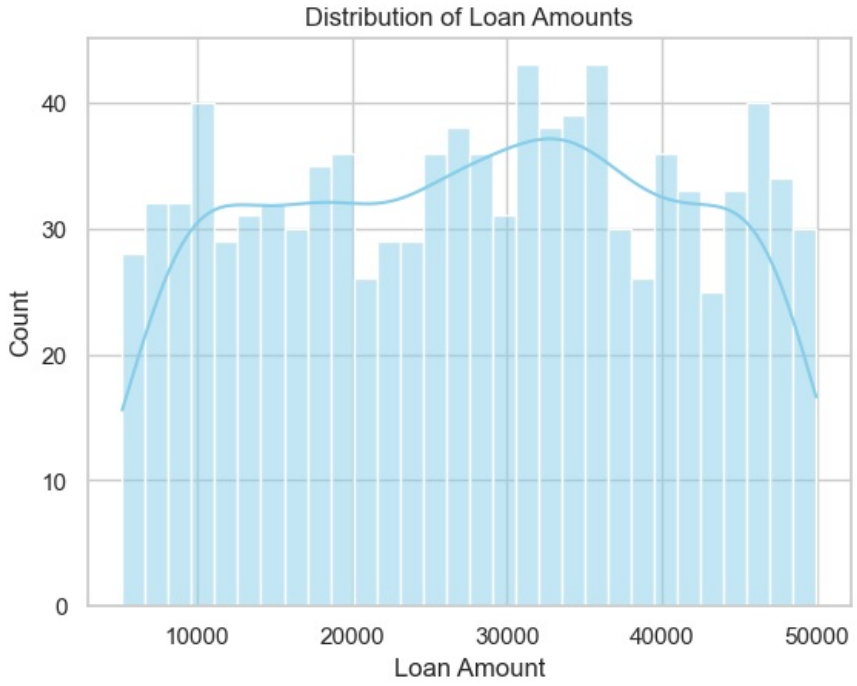
Loan Status Distribution

In [21]:
```
status_counts = fact_loan['Status'].value_counts()
sns.countplot(data=fact_loan, x='Status', palette='Set2')
plt.title("Loan Status Distribution")
plt.show()
```



Loan Amount Distribution

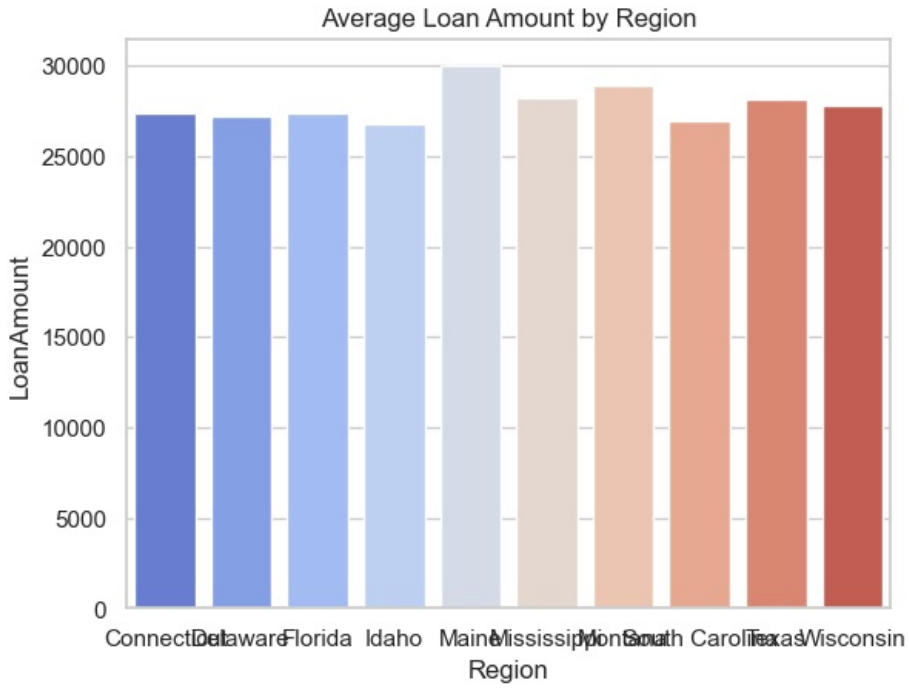
In [22]:
```
sns.histplot(fact_loan['LoanAmount'], bins=30, kde=True, color='skyblue')
plt.title("Distribution of Loan Amounts")
plt.xlabel("Loan Amount")
plt.show()
```



Average Loan by Region

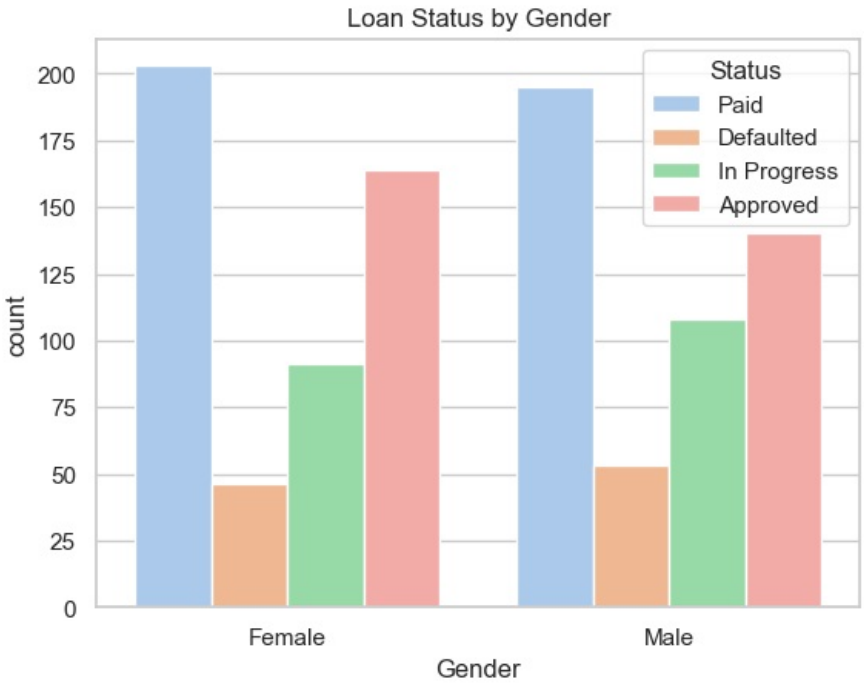
In [23]:
```
merged = fact_loan.merge(dim_branch, on='BranchID')
region_avg = merged.groupby('Region')['LoanAmount'].mean().reset_index()

sns.barplot(data=region_avg, x='Region', y='LoanAmount', palette='coolwarm')
plt.title("Average Loan Amount by Region")
plt.show()
```



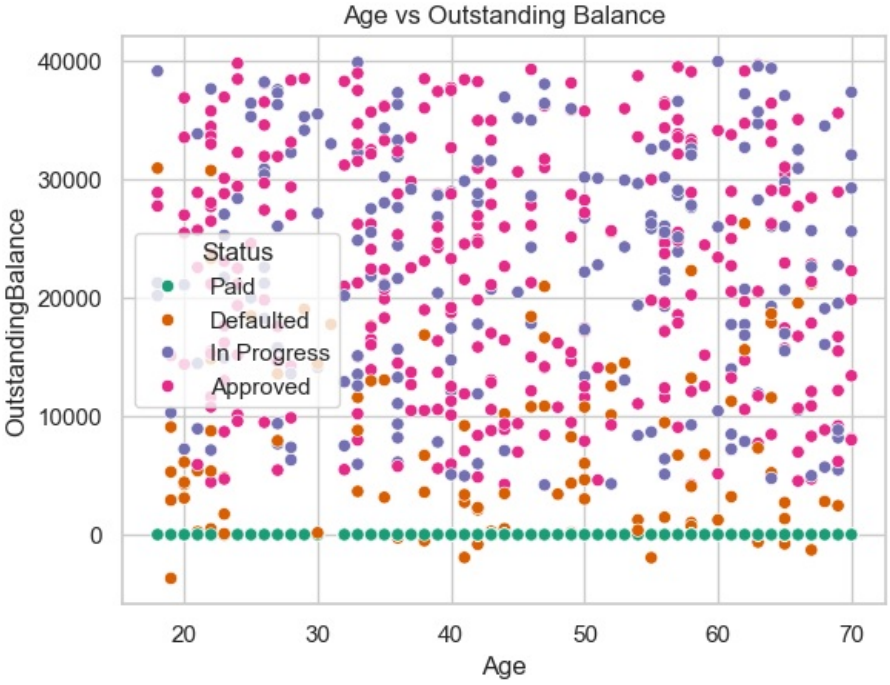
Loan Performance by Gender

In [24]:
```
merged = fact_loan.merge(dim_customer, on='CustomerID')
sns.countplot(data=merged, x='Gender', hue='Status', palette='pastel')
plt.title("Loan Status by Gender")
plt.show()
```



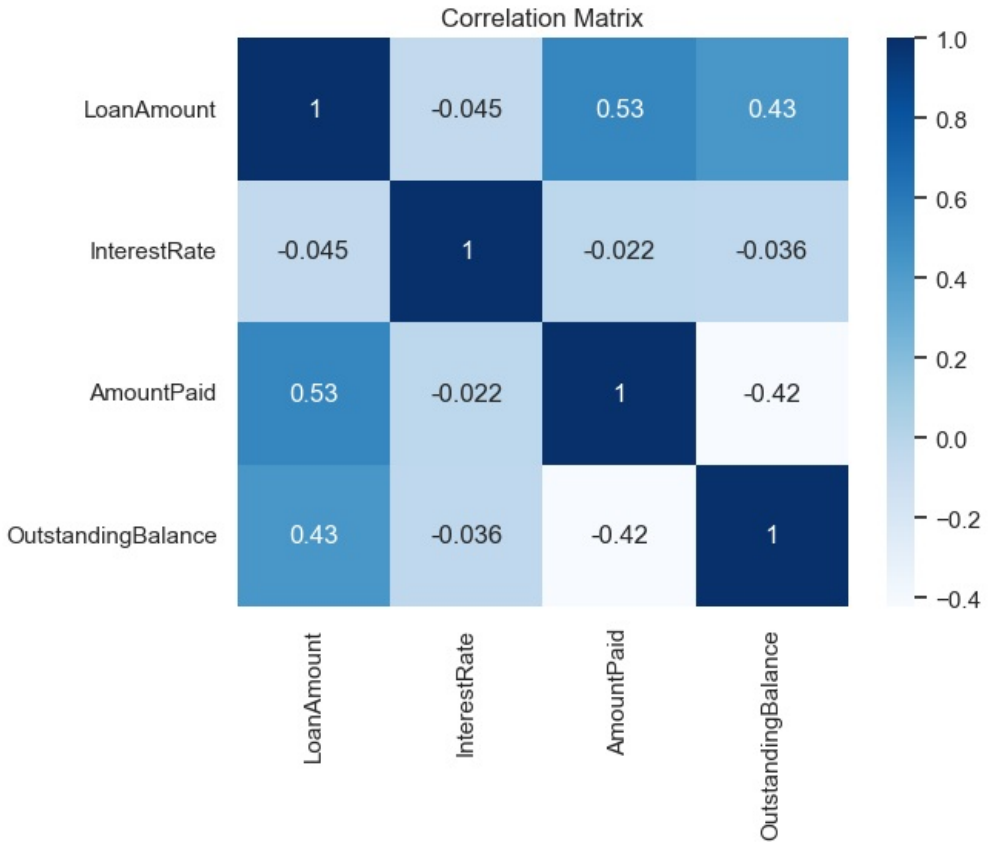
Customer Age vs Outstanding Balance

In [26]:
```
sns.scatterplot(data=merged, x='Age', y='OutstandingBalance', hue='Status', palette='Dark2')
plt.title("Age vs Outstanding Balance")
plt.show()
```



Correlation Heatmap (Numerical)

In [27]:
```
corr = fact_loan[['LoanAmount', 'InterestRate', 'AmountPaid', 'OutstandingBalance']].corr()
sns.heatmap(corr, annot=True, cmap='Blues')
plt.title("Correlation Matrix")
plt.show()
```



In [ ]:

In [ ]: