```
import pandas as pd
import numpy as npdf
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df = pd.read_csv(r'C:\Users\USER\Downloads\employee_survey.csv', sep =',')
df
```

	EmpID	Gender	Age	MaritalStatus	JobLevel	Experience	Dept	EmpType	WLB	WorkEnv
0	6	Male	32	Married	Mid	7	IT	Full-Time	1	1
1	11	Female	34	Married	Mid	12	Finance	Full-Time	1	1
2	33	Female	23	Single	Intern/Fresher	1	Marketing	Full-Time	2	4
3	20	Female	29	Married	Junior	6	IT	Contract	2	2
4	28	Other	23	Single	Junior	1	Sales	Part-Time	3	1
...	...	...	...	...	...	...	...	...	...	...
3020	2070	Male	47	Married	Lead	16	Marketing	Part-Time	1	3
3021	2072	Female	47	Married	Lead	8	IT	Contract	3	1
3022	2145	Male	41	Married	Lead	17	IT	Full-Time	1	1
3023	2168	Female	54	Married	Lead	16	IT	Full-Time	1	5
3024	2183	Female	44	Married	Lead	22	Marketing	Full-Time	3	3

3025 rows × 23 columns

```
df.shape
```

```
(3025, 23)
```

```
total = df.isnull().sum().sort_values(ascending= False)
total
```

```
EmpID                    0
Stress                   0
TrainingHoursPerYear     0
haveOT                   0
EduLevel                 0
NumReports               0
TeamSize                 0
NumCompanies             0
CommuteDistance          0
CommuteMode              0
SleepHours               0
Workload                 0
Gender                   0
PhysicalActivityHours    0
WorkEnv                  0
WLB                      0
EmpType                  0
Dept                     0
Experience               0
JobLevel                 0
MaritalStatus            0
```

```
Age                       0
JobSatisfaction           0
dtype: int64
```

```
df = df[['WLB', 'JobSatisfaction', 'Stress', 'WorkEnv']]

# Memeriksa data untuk memastikan tidak ada nilai NaN
print(df.isnull().sum())
```

```
WLB                0
JobSatisfaction    0
Stress             0
WorkEnv            0
dtype: int64
```

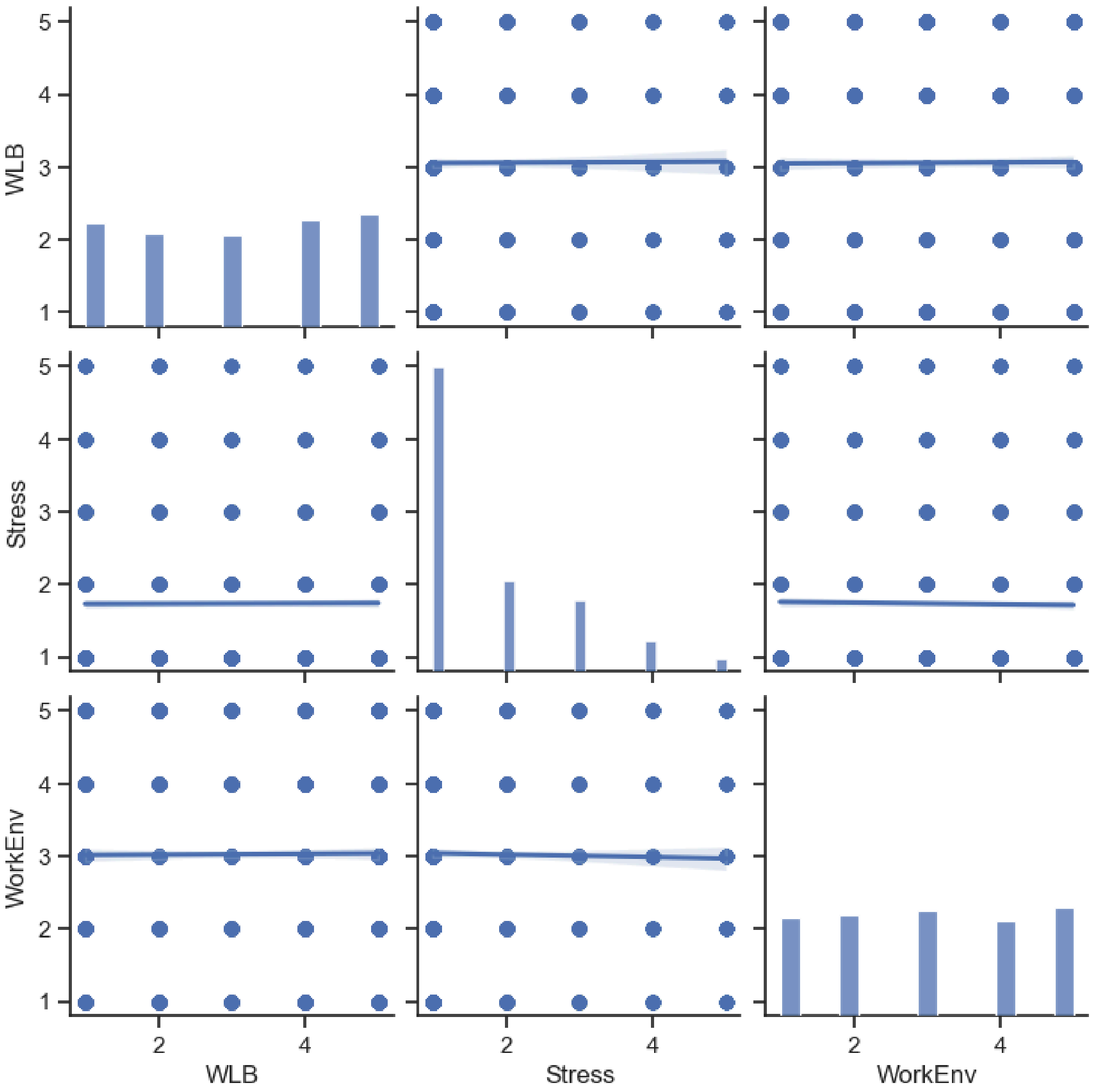
```
WLB
```

```
Series([], Name: JobSatisfaction, dtype: int64)
```

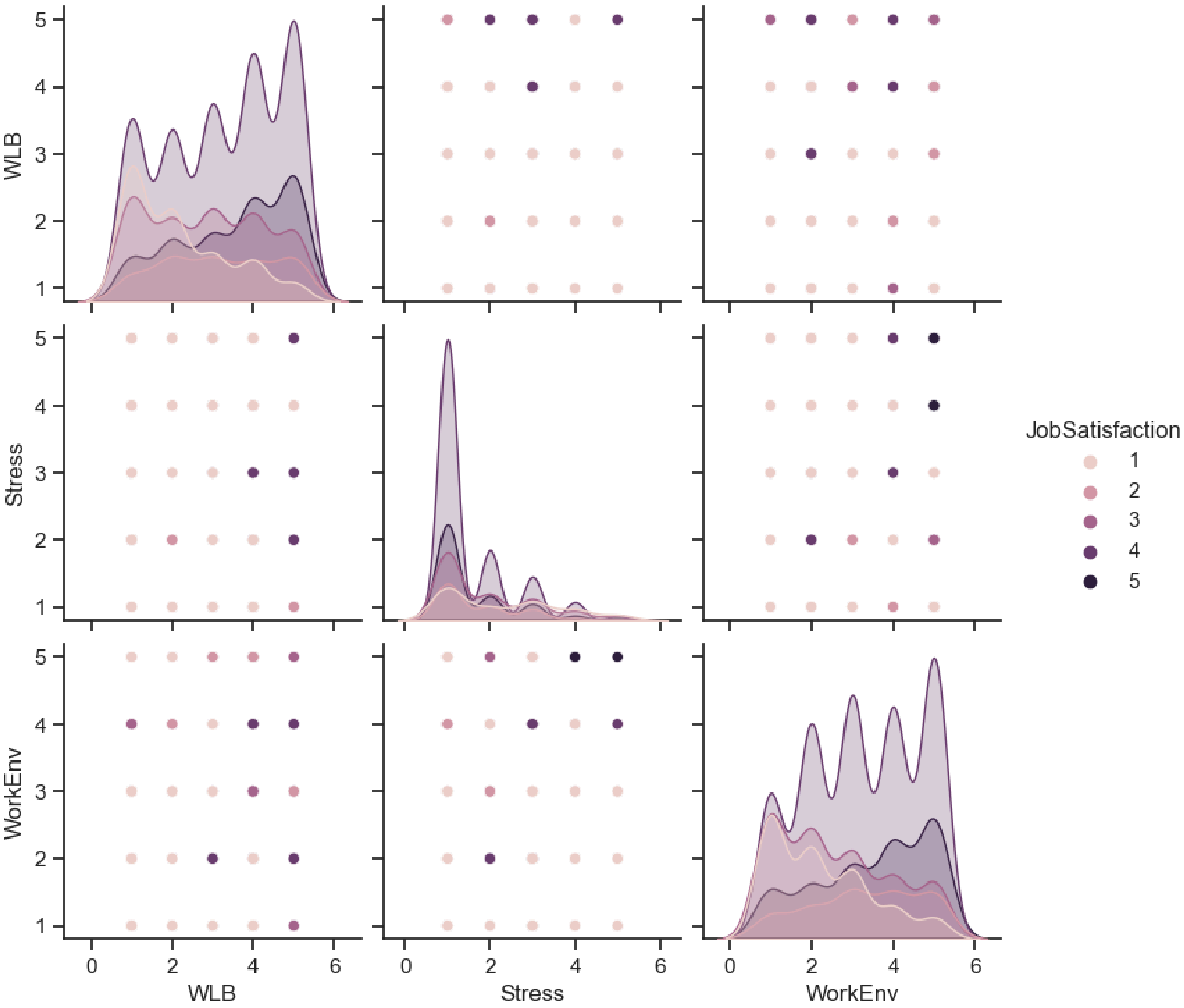
```
# Mendefinisikan variabel independen dan dependen
import statsmodels.api as sm
X = df[['WLB', 'Stress', 'WorkEnv']]
y = df['JobSatisfaction']

# Menambahkan konstanta ke model (intercept)
X = sm.add_constant(X)

# Membuat model regresi linier
model = sm.OLS(y, X).fit()

# Melihat hasil regresi
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        JobSatisfaction   R-squared:                       0.173
Model:                            OLS   Adj. R-squared:                  0.173
Method:                 Least Squares   F-statistic:                     211.4
Date:                Sat, 09 Nov 2024   Prob (F-statistic):          1.89e-124
Time:                        12:23:28   Log-Likelihood:                -4704.2
No. Observations:                3025   AIC:                             9416.
Df Residuals:                    3021   BIC:                             9440.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.4727      0.074     33.227      0.000       2.327       2.619
WLB            0.2257      0.014     15.661      0.000       0.197       0.254
Stress        -0.2510      0.020    -12.812      0.000      -0.289      -0.213
WorkEnv        0.2176      0.015     14.769      0.000       0.189       0.246
==============================================================================
Omnibus:                      144.291   Durbin-Watson:                   0.264
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              123.154
Skew:                          -0.423   Prob(JB):                     1.81e-27
Kurtosis:                       2.489   Cond. No.                         18.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
# pair plot with plot type regression
sns.pairplot(df,vars = ['WLB', 'Stress','WorkEnv'], kind="reg")
plt.show()
```



```
 #pair plot (matrix scatterplot) of few columns
sns.set(style="ticks", color_codes=True)
sns.pairplot(df,vars = ['WLB', 'Stress','WorkEnv'], hue="JobSatisfaction")
plt.show()
```



```
from scipy import stats

# Menghitung korelasi Pearson untuk masing-masing variabel terhadap JobSatisfaction
variables = ['WLB', 'Stress', 'WorkEnv']
for var in variables:
    corr = stats.pearsonr(df[var], df['JobSatisfaction'])
    print(f"Korelasi antara {var} dan JobSatisfaction:")
    print("p-value:\t", corr[1])
    print("cor:\t\t", corr[0])
    print()
```

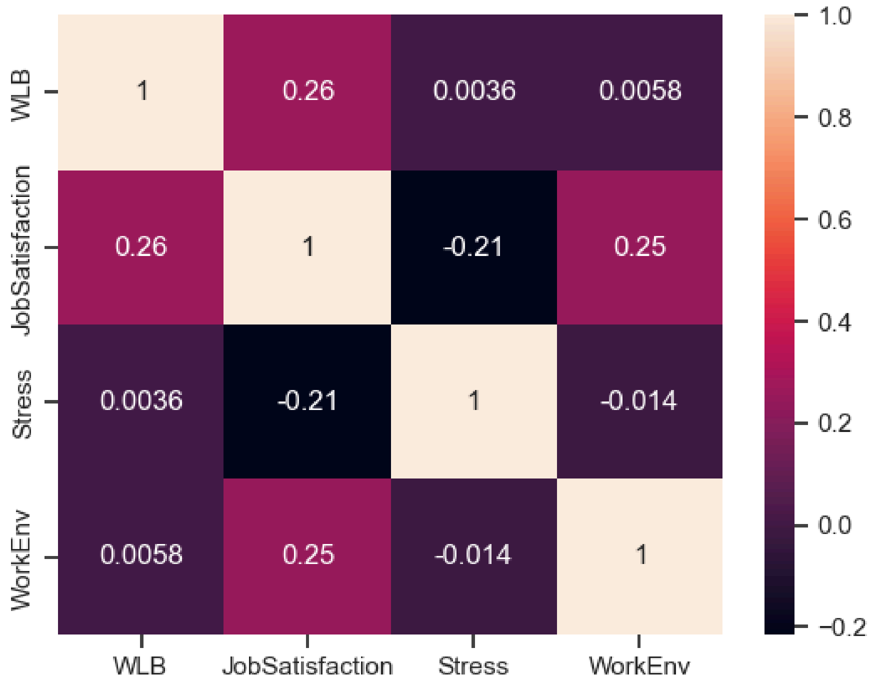
```
Korelasi antara WLB dan JobSatisfaction:
p-value:         8.194092905974926e-48
cor:             0.2596872800125278

Korelasi antara Stress dan JobSatisfaction:
p-value:         8.759834103275332e-33
cor:             -0.21438269956302416

Korelasi antara WorkEnv dan JobSatisfaction:
p-value:         6.945939004062785e-44
cor:             0.2487300192722577
```

```
correlation = df.corr(method='pearson')
correlation
```

	WLB	JobSatisfaction	Stress	WorkEnv
WLB	1.000000	0.259687	0.003595	0.005751
JobSatisfaction	0.259687	1.000000	-0.214383	0.248730
Stress	0.003595	-0.214383	1.000000	-0.013791
WorkEnv	0.005751	0.248730	-0.013791	1.000000

```
sns.heatmap(correlation,xticklabels=correlation.columns, yticklabels=correlation.columns, annot=Tr
```

```
<Axes: >
```



Start coding or generate with AI.