NOTEBOOK DESTINADO A REGISTRAR AS TAREFAS DA DISCIPLINA DE AEDI - 1º/2025

PROFESSOR: JOÃO GABRIEL DE MORAES SOUZA

ALUNO: GUSTAVO PARREIRA LIMA CUNHA

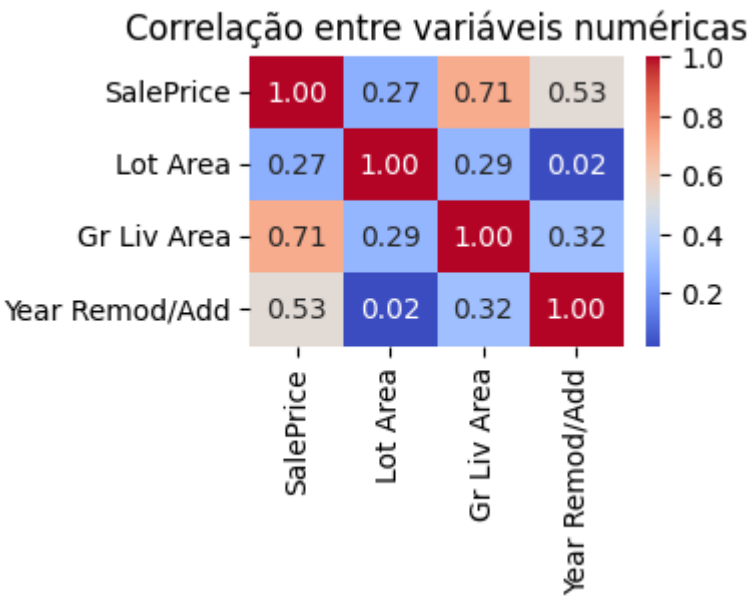
TAREFA 4 - REGRESSÃO LINEAR

Questão A - Modelagem do preço de venda

In [1]:
```
import kagglehub
import pandas as pd
import os
import matplotlib.pyplot as plt
import seaborn as sns

# Baixar dataset
path = kagglehub.dataset_download("prevek18/ames-housing-dataset")
print("Path to dataset files:", path)

# leitura do arquivo
csv_path = os.path.join(path, "AmesHousing.csv")
df = pd.read_csv(csv_path)

# Análise exploratória: explorar algumas variáveis e como elas se relacionam com o

print(df.info())

# Correlação numérica entre as variáveis

plt.subplot(2, 2, 4)
correlation = df[["SalePrice", "Lot Area", "Gr Liv Area", "Year Remod/Add"]].corr(
sns.heatmap(correlation, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlação entre variáveis numéricas")

plt.tight_layout()
plt.show()
```

```
Path to dataset files: /kaggle/input/ames-housing-dataset
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 82 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Order            2930 non-null   int64
 1   PID              2930 non-null   int64
 2   MS SubClass      2930 non-null   int64
 3   MS Zoning        2930 non-null   object
 4   Lot Frontage     2440 non-null   float64
 5   Lot Area         2930 non-null   int64
 6   Street           2930 non-null   object
 7   Alley            198 non-null    object
 8   Lot Shape        2930 non-null   object
 9   Land Contour     2930 non-null   object
 10  Utilities        2930 non-null   object
 11  Lot Config       2930 non-null   object
 12  Land Slope       2930 non-null   object
 13  Neighborhood     2930 non-null   object
 14  Condition 1      2930 non-null   object
 15  Condition 2      2930 non-null   object
 16  Bldg Type        2930 non-null   object
 17  House Style      2930 non-null   object
 18  Overall Qual     2930 non-null   int64
 19  Overall Cond     2930 non-null   int64
 20  Year Built       2930 non-null   int64
 21  Year Remod/Add   2930 non-null   int64
 22  Roof Style       2930 non-null   object
 23  Roof Matl        2930 non-null   object
 24  Exterior 1st     2930 non-null   object
 25  Exterior 2nd     2930 non-null   object
 26  Mas Vnr Type     1155 non-null   object
 27  Mas Vnr Area     2907 non-null   float64
 28  Exter Qual       2930 non-null   object
 29  Exter Cond       2930 non-null   object
 30  Foundation       2930 non-null   object
 31  Bsmt Qual        2850 non-null   object
 32  Bsmt Cond        2850 non-null   object
 33  Bsmt Exposure    2847 non-null   object
 34  BsmtFin Type 1   2850 non-null   object
 35  BsmtFin SF 1     2929 non-null   float64
 36  BsmtFin Type 2   2849 non-null   object
 37  BsmtFin SF 2     2929 non-null   float64
 38  Bsmt Unf SF      2929 non-null   float64
 39  Total Bsmt SF    2929 non-null   float64
 40  Heating          2930 non-null   object
 41  Heating QC       2930 non-null   object
 42  Central Air      2930 non-null   object
 43  Electrical       2929 non-null   object
 44  1st Flr SF       2930 non-null   int64
 45  2nd Flr SF       2930 non-null   int64
 46  Low Qual Fin SF  2930 non-null   int64
 47  Gr Liv Area      2930 non-null   int64
 48  Bsmt Full Bath   2928 non-null   float64
 49  Bsmt Half Bath   2928 non-null   float64
 50  Full Bath        2930 non-null   int64
 51  Half Bath        2930 non-null   int64
 52  Bedroom AbvGr    2930 non-null   int64
 53  Kitchen AbvGr    2930 non-null   int64
```

```
 54  Kitchen Qual     2930 non-null   object
 55  TotRms AbvGrd    2930 non-null   int64
 56  Functional       2930 non-null   object
 57  Fireplaces       2930 non-null   int64
 58  Fireplace Qu     1508 non-null   object
 59  Garage Type      2773 non-null   object
 60  Garage Yr Blt    2771 non-null   float64
 61  Garage Finish    2771 non-null   object
 62  Garage Cars      2929 non-null   float64
 63  Garage Area      2929 non-null   float64
 64  Garage Qual      2771 non-null   object
 65  Garage Cond      2771 non-null   object
 66  Paved Drive      2930 non-null   object
 67  Wood Deck SF     2930 non-null   int64
 68  Open Porch SF    2930 non-null   int64
 69  Enclosed Porch   2930 non-null   int64
 70  3Ssn Porch       2930 non-null   int64
 71  Screen Porch     2930 non-null   int64
 72  Pool Area        2930 non-null   int64
 73  Pool QC          13 non-null     object
 74  Fence            572 non-null    object
 75  Misc Feature     106 non-null    object
 76  Misc Val         2930 non-null   int64
 77  Mo Sold          2930 non-null   int64
 78  Yr Sold          2930 non-null   int64
 79  Sale Type        2930 non-null   object
 80  Sale Condition   2930 non-null   object
 81  SalePrice        2930 non-null   int64
dtypes: float64(11), int64(28), object(43)
memory usage: 1.8+ MB
None
```



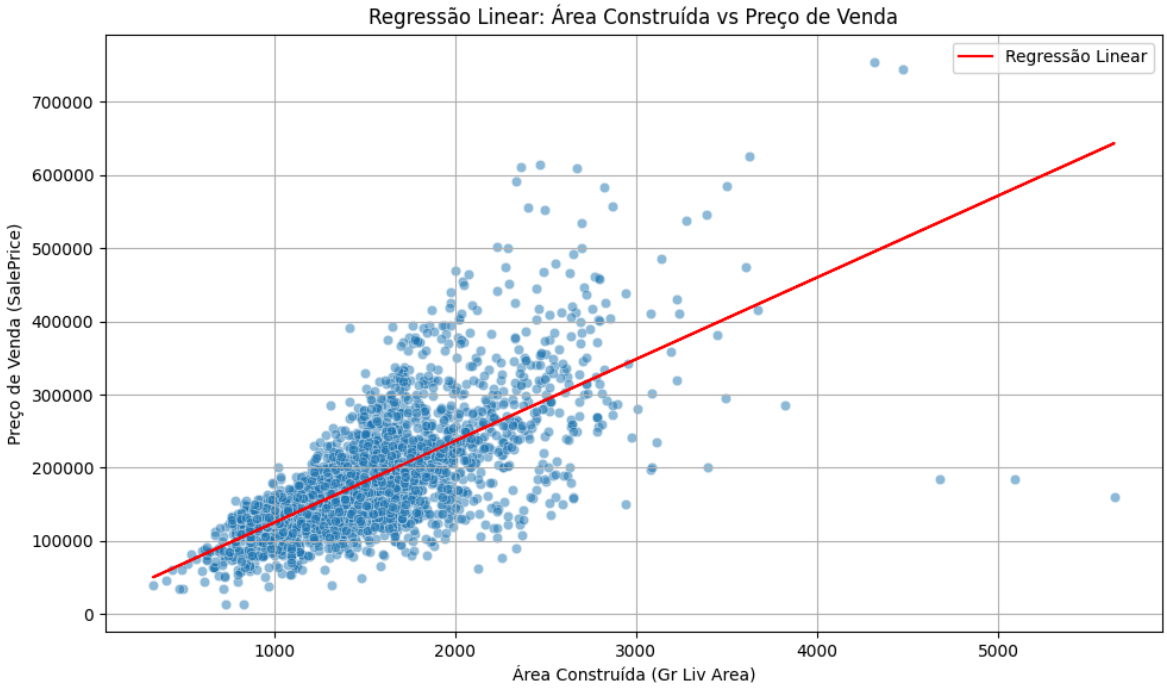
A partir da matriz de correlação entre as variáveis, foi constatado que, contrariamente à hipótese de que o preço de venda era altamente impactado pela área do lote, essas variáveis não se mostraram fortemente correlacionadas. Portanto, será levada adiante a análise somente entre o preço de venda e a área construída (Gr Liv Area) e o ano de reforma ou ampliação (Year Remod/Add).

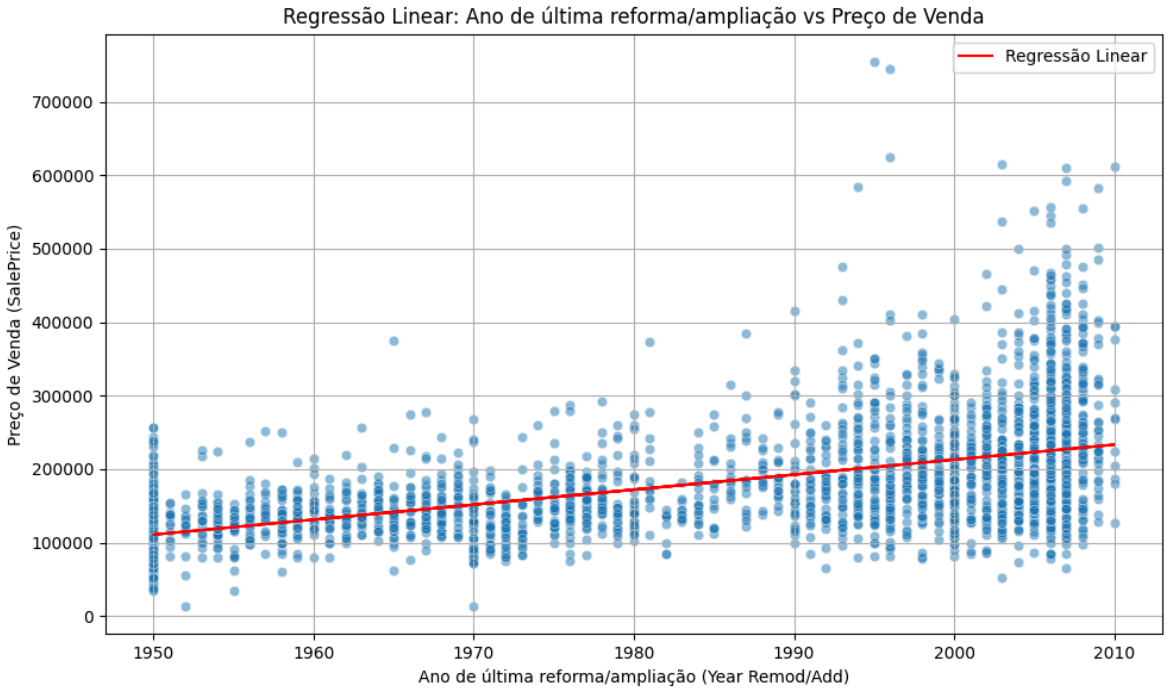
In [2]:
```
from sklearn.linear_model import LinearRegression
import numpy as np
```

```
# tratamento de dados
# Remover valores nulos
df_clean = df[["Gr Liv Area", "SalePrice"]].dropna()
df_clean2 = df[["Year Remod/Add", "SalePrice"]].dropna()

# Variáveis
X1 = df_clean[["Gr Liv Area"]] #variável explicativa
y1 = df_clean["SalePrice"] # variável dependente

X2 = df_clean2[["Year Remod/Add"]] #variável explicativa

# Modelo de regressão
model1 = LinearRegression()
model1.fit(X1, y1)
y_pred1 = model1.predict(X1)

model2 = LinearRegression()
model2.fit(X2, y1)
y_pred2 = model2.predict(X2)

# Coeficientes
intercept = model1.intercept_
slope = model1.coef_[0]
r2 = model1.score(X1, y1)

intercept2 = model2.intercept_
slope2 = model2.coef_[0]
r22 = model2.score(X2, y1)

print(f"Coeficiente linear (Área construída): {intercept:.2f}")
print(f"Coeficiente angular (Área construída): {slope:.2f}")
print()
print(f"Coeficiente linear (Ano de última reforma/ampliação): {intercept2:.2f}")
print(f"Coeficiente angular (Ano de última reforma/ampliação): {slope2:.2f}")

# Gráfico 1
plt.figure(figsize=(10, 6))
sns.scatterplot(x="Gr Liv Area", y="SalePrice", data=df_clean, alpha=0.5)
plt.plot(X1, y_pred1, color='red', label='Regressão Linear')

plt.title("Regressão Linear: Área Construída vs Preço de Venda")
plt.xlabel("Área Construída (Gr Liv Area)")
plt.ylabel("Preço de Venda (SalePrice)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
print()
#gráfico 2
plt.figure(figsize=(10, 6))
sns.scatterplot(x="Year Remod/Add", y="SalePrice", data=df_clean2, alpha=0.5)
plt.plot(X2, y_pred2, color='red', label='Regressão Linear')

plt.title("Regressão Linear: Ano de última reforma/ampliação vs Preço de Venda")
plt.xlabel("Ano de última reforma/ampliação (Year Remod/Add)")
plt.ylabel("Preço de Venda (SalePrice)")
plt.legend()
plt.grid(True)
```

```
plt.tight_layout()
plt.show()
```

```
Coeficiente linear (Área construída): 13289.63
Coeficiente angular (Área construída): 111.69

Coeficiente linear (Ano de última reforma/ampliação): -3869250.53
Coeficiente angular (Ano de última reforma/ampliação): 2041.08
```





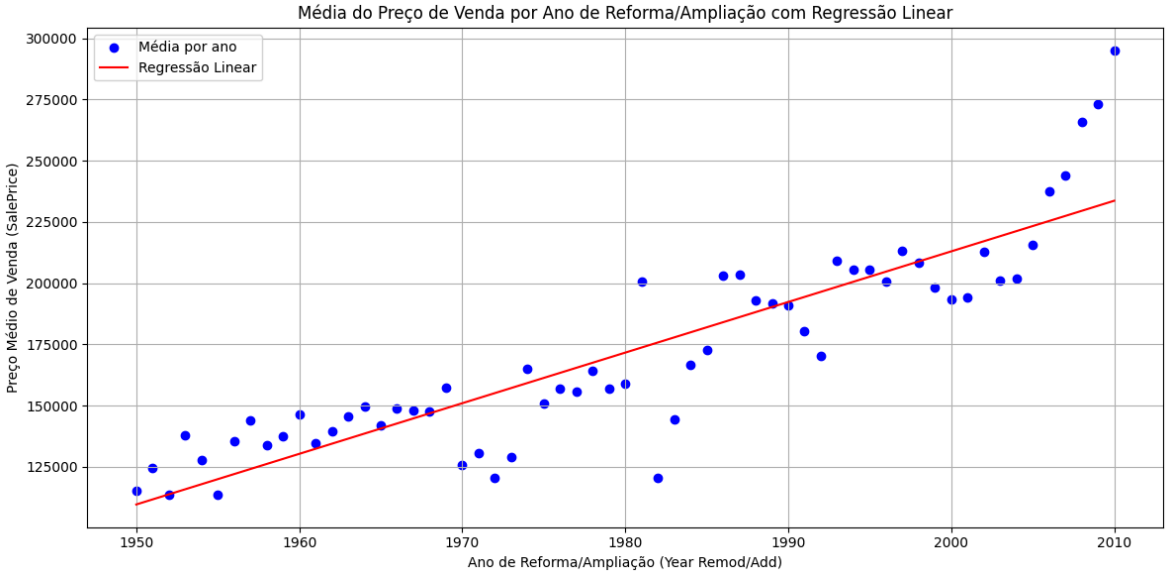
Aparentemente, sob o aspecto visual, o gráfico da regressão linear indica que a variável "ano de última reforma/ampliaçao", apesar de apresentar boa correlação com o preço de venda, não é boa para gerar um modelo de previsão de preço de venda a partir dela, especialmente por ter valores discretos (anos). Para testar isso, será aplicado o coeficiente de determinação, que avalia o quão bem o modelo de regressão consegue prever ou explicar os valores observados para uma dada variável.

In [3]:
```
from sklearn.linear_model import LinearRegression

print(f"R² da regressão da variável explicativa ano de última reforma/ampliação: {

print(f"R² da regressão da variável explicativa área construída: {r2:.4f}")
```

```
R² da regressão da variável explicativa ano de última reforma/ampliação: 0.2841
R² da regressão da variável explicativa área construída: 0.4995
```

O valor do coeficiente de determinação obtido foi significativo R^2 = 0,2841, o que indica que, apesar de visualmente a regressão apresentar comportamento questionável, a variável é numericamente relevante para a previsão do preço de venda. Portanto, manter-se-á sua utilização. O gráfico abaixo, que compara a média do preço de venda das casas de acordo com o ano de última reforma/ampliaçao apresenta uma melhor visualização da relação entre as duas variáveis. A regressão, quando analisada isoladamente, apresentou um resultado visualmente frágil para mostrar essa relação, tendo sido necessário portanto aprofundar a investigação da variável. Com esse aprofundamento, que se pautou no cálculo do coeficiente de determinação, foi possível afirmar seguramente a relação entre as duas variáveis.

In [4]:
```
# Agrupar por ano e calcular a média
media_por_ano = df.groupby('Year Remod/Add')['SalePrice'].mean().reset_index()

# Valores para regressão
X3 = media_por_ano['Year Remod/Add'].values.reshape(-1, 1)
y3 = media_por_ano['SalePrice'].values

# Regressão linear
model3 = LinearRegression()
model3.fit(X3, y3)
y_pred3 = model3.predict(X3)

# Plotar pontos e regressão
plt.figure(figsize=(12, 6))
plt.scatter(media_por_ano['Year Remod/Add'], media_por_ano['SalePrice'], color='bl
plt.plot(media_por_ano['Year Remod/Add'], y_pred3, color='red', label='Regressão l
plt.title('Média do Preço de Venda por Ano de Reforma/Ampliação com Regressão Line
plt.xlabel('Ano de Reforma/Ampliação (Year Remod/Add)')
plt.ylabel('Preço Médio de Venda (SalePrice)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



Análise dos coeficientes encontrados:

Regressão para a variável Área Construída Intercepto: 13289.63 Coeficiente angular: 111.69

Regressão para a variável Ano de última reforma/ampliação Intercepto: -3869250.53 Coeficiente angular: 2041.08

O valor do coeficiente angular de cada curva indica o impacto que a variável explicativa tem na determinação do preço de venda(variável dependente). Ou seja, quanto maior a inclinação, mais o preço de venda será impactado por uma eventual variação da variável explicativa.

No caso da área construída, a cada 1 pé quadrado a mais de área construída, o preço de venda aumenta, em média US 111,69. Da mesma forma, para cada ano mais recente de reforma/ampliação, o valor da casa aumenta em média US 2.041,08.

Ao se analisar os coeficientes de determinação: R^2 da regressão da variável explicativa ano de última reforma/ampliação: 0.2841 R^2 da regressão da variável explicativa área construída: 0.4995

Nota-se que a área construída explica aproximadamente 50% do incremento do preço das casas, contra 28% explicados pelo ano de última intervenção civil na edificação.

Questão B - Validação dos Pressupostos da Regressão Linear

1- Os coeficientes β1 e β2 devem ser lineares.

O gráfico que representa a regressão linear do preço de venda de acordo com a ***área construída*** apresenta visualmente comportamento linear. Por outro lado, a regressão da variável preço médio de venda por ***ano de última reforma*** indica comportamento não

linear para os valores finais da variável explicativa, em especial após 2005. Portanto, será considerado que B1 e B2 da segunda variável explicativa não são lineares.

Não sendo lineares, não seria possível prever o preço de venda de uma casa a partir da aplicação de regressão linear na variável "ano de última reforma".

2- O Valor Esperado de ui é zero

In [5]:
```
# Resíduos (erros)
residuos1 = y1 - y_pred1

residuos2 = y1 - y_pred2

# Soma dos resíduos
soma_residuos1 = residuos1.sum()

soma_residuos2 = residuos2.sum()

print("Soma dos resíduos 1:", soma_residuos1)
print()
print("Soma dos resíduos 2:", soma_residuos2)
```

```
Soma dos resíduos 1: -4.237517714500427e-08

Soma dos resíduos 2: -4.4656917452812195e-07
```

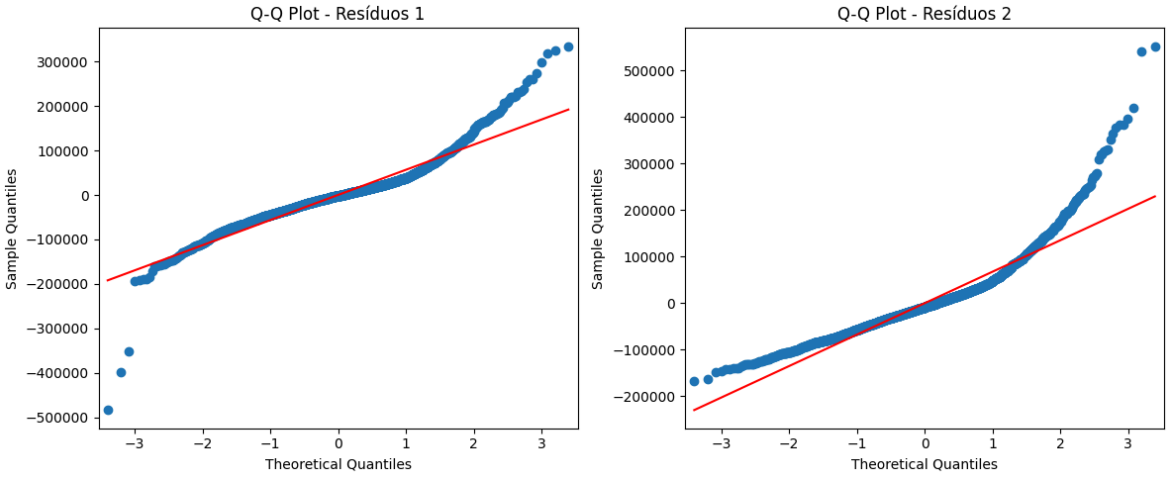
Os resíduos apresentaram valor tendendo a zero, portanto, as variáveis atendem a esse requisito.

Teste da normalidade dos resíduos

In [6]:
```
from scipy.stats import shapiro

# Teste de normalidade dos resíduos
stat1, p1 = shapiro(residuos1)
stat2, p2 = shapiro(residuos2)

print(f"Resíduos 1 - Estatística: {stat1:.4f}, p-valor: {p1:.4f}")
print(f"Resíduos 2 - Estatística: {stat2:.4f}, p-valor: {p2:.4f}")

# Interpretação
if p1 < 0.05:
    print("Resíduos 1: rejeita-se a normalidade (p < 0.05)")
else:
    print("Resíduos 1: não se rejeita a normalidade")

if p2 < 0.05:
    print("Resíduos 2: rejeita-se a normalidade (p < 0.05)")
else:
    print("Resíduos 2: não se rejeita a normalidade")


import matplotlib.pyplot as plt
import statsmodels.api as sm

# Q-Q Plot para resíduos 1
```

```
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
sm.qqplot(residuos1, line='s', ax=plt.gca())
plt.title('Q-Q Plot - Resíduos 1')

# Q-Q Plot para resíduos 2
plt.subplot(1, 2, 2)
sm.qqplot(residuos2, line='s', ax=plt.gca())
plt.title('Q-Q Plot - Resíduos 2')

plt.tight_layout()
plt.show()
```

```
Resíduos 1 - Estatística: 0.9211, p-valor: 0.0000
Resíduos 2 - Estatística: 0.8944, p-valor: 0.0000
Resíduos 1: rejeita-se a normalidade (p < 0.05)
Resíduos 2: rejeita-se a normalidade (p < 0.05)
```

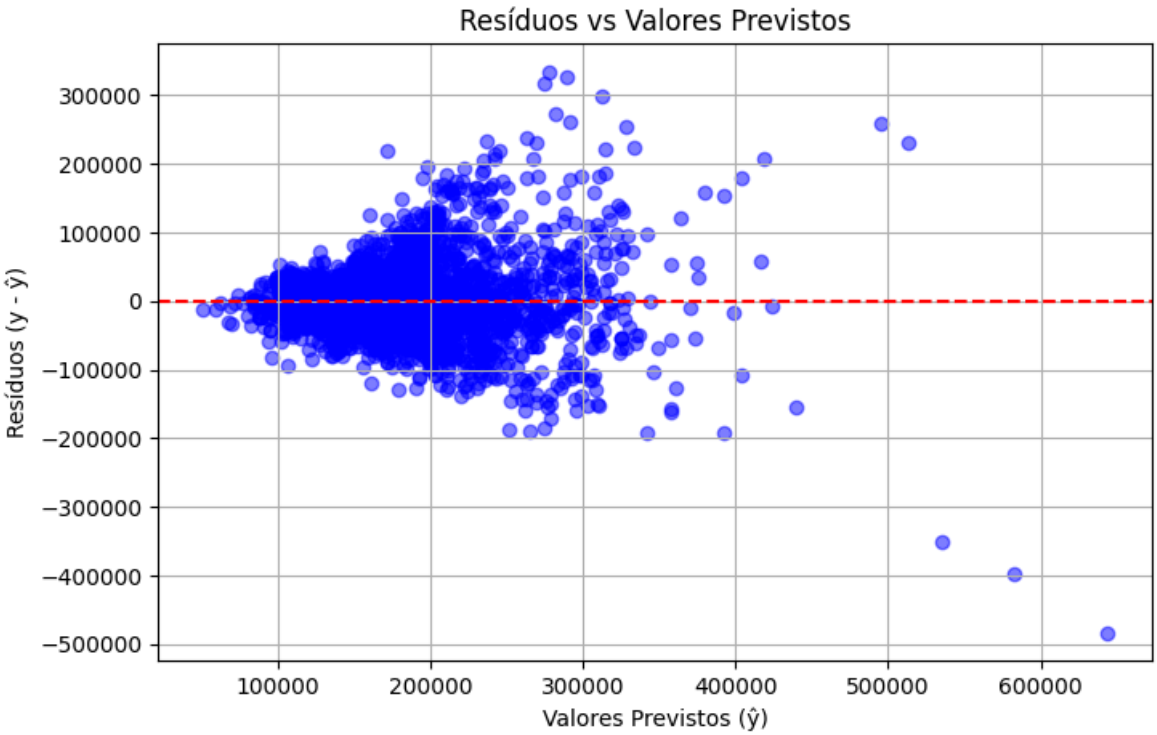


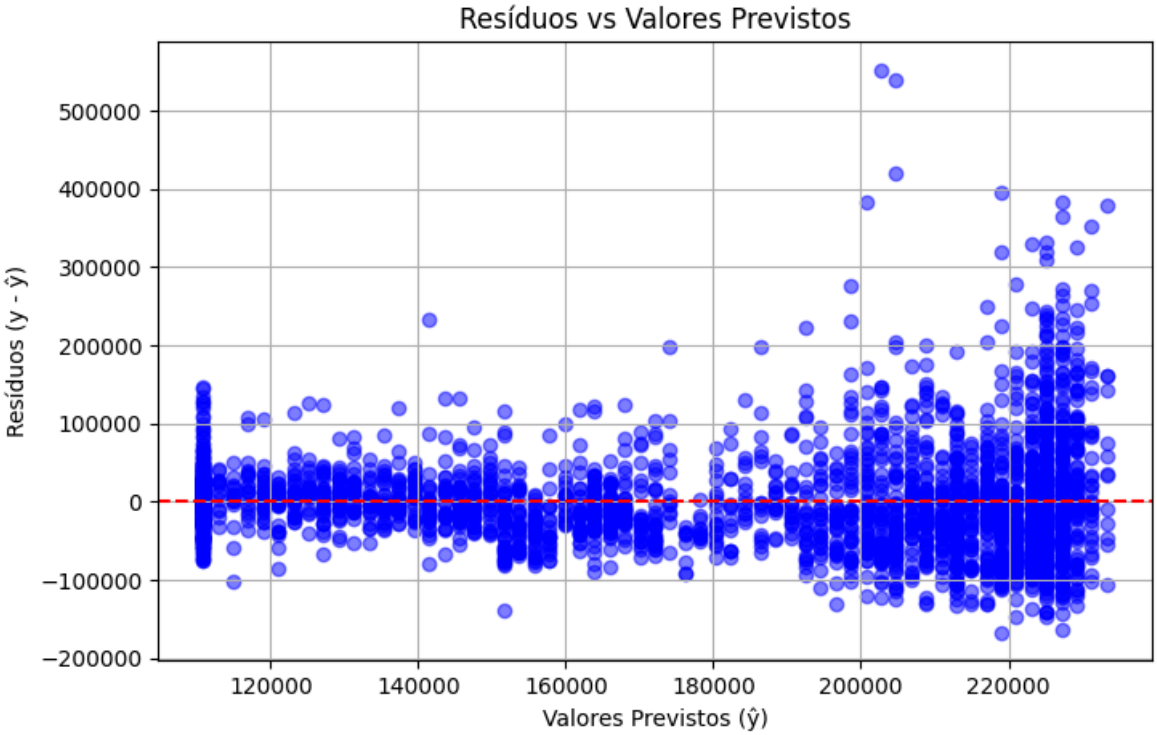
A partir da aplicação do teste de Shapiro e da curva QQ-Plot, observa-se que os resíduos não se distribuem conforme uma normal. Portanto não atende a essa premissa.

3- Homocedasticidade ou variância igual de ui

In [7]:
```
# teste de variância de ui (resíduos)

# INSPEÇÃO VISUAL DOS RESÍDUOS (para testar se variância é constante para todos os

# Plotando resíduos vs valores previstos
plt.figure(figsize=(8, 5))
plt.scatter(y_pred1, residuos1, color='blue', alpha=0.5)
plt.axhline(y=0, color='red', linestyle='--')
plt.xlabel('Valores Previstos (ŷ)')
plt.ylabel('Resíduos (y - ŷ)')
plt.title('Resíduos vs Valores Previstos')
plt.grid(True)
plt.show()



# Plotando resíduos vs valores previstos
plt.figure(figsize=(8, 5))
plt.scatter(y_pred2, residuos2, color='blue', alpha=0.5)
plt.axhline(y=0, color='red', linestyle='--')
```

```
plt.xlabel('Valores Previstos (ŷ)')
plt.ylabel('Resíduos (y - ŷ)')
plt.title('Resíduos vs Valores Previstos')
plt.grid(True)
plt.show()
```





Inspeção visual indicou variância variável para valores crescentes das duas variáveis explicativas. Portanto, será executado um teste numérico de Breusch-Pagan para confirmar se de fato são ou não constantes os resíduos.

In [8]:
```
import statsmodels.api as sm
from statsmodels.stats.diagnostic import het_breuschpagan

# y: variável dependente e X: variável independente(s)
```

```
# Adicionando constante à matriz de variáveis independentes
X1m = sm.add_constant(X1)  # necessário para o modelo com intercepto
X2m = sm.add_constant(X2)
# Ajusta o modelo
modelo1 = sm.OLS(y1, X1m).fit()
modelo2 = sm.OLS(y1, X2m).fit()
# Obtém os resíduos
residuos1 = modelo1.resid
residuos2 = modelo2.resid
# Teste de Breusch-Pagan
# A função espera (resíduos, matriz X do modelo original)
teste1 = het_breuschpagan(residuos1, X1m)
teste2 = het_breuschpagan(residuos2, X2m)
# Resultado do teste
labels = ['LM Statistic', 'LM p-value', 'F-Statistic', 'F p-value']
resultado1 = dict(zip(labels, teste1))
resultado2 = dict(zip(labels, teste2))
# Mostra os resultados
for k, v in resultado1.items():
    print(f'{k}: {v:.4f}')
for k, v in resultado2.items():
    print(f'{k}: {v:.4f}')
```

```
LM Statistic: 592.3920
LM p-value: 0.0000
F-Statistic: 742.0080
F p-value: 0.0000
LM Statistic: 123.9328
LM p-value: 0.0000
F-Statistic: 129.3181
F p-value: 0.0000
```

Como LM p-value e F p-value obtidos foram praticamente nulos, tem-se a rejeição da hipótese nula, ou seja, em ambos os casos rejeita-se a hipótese nula de homocedasticidade, **recaindo-se em heterocedasticidade**. Ou seja, a variância dos resíduos não é constante em todos os níveis da variável explicativa. Viola-se a premissa básica para aplicação da regressão linear.

Como último passo para decisão, será verificada a multicolinearidade entre as variáveis, dado pelo índice "VIF". Um VIF até 10 indica que das variáveis explicativas estão pouco relcionadas, e pode-se utiliza-se da regressão para previsão e/ou explicação sobre elas.

In [9]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Suponha que df seja seu DataFrame com as variáveis
# Se necessário, substitua por seu DataFrame real

Xvif = df[['Year Remod/Add', 'Gr Liv Area']].dropna()
Xvif = sm.add_constant(Xvif)

# Calcular o VIF para cada variável
vif_data = pd.DataFrame()
vif_data['Variável'] = Xvif.columns
vif_data['VIF'] = [variance_inflation_factor(Xvif.values, i) for i in range(Xvif.s

print(vif_data)
```

```
         Variável          VIF
0           const  9873.294107
1  Year Remod/Add     1.111601
2     Gr Liv Area     1.111601
```

Não foi encontrada multicolinearidade entre as variáveis explicativas, portanto os problemas que impedem a representação do fenômeno por uma regressão linear são:

1. Não normalidade dos resíduos,
2. Heterocedasticidade dos resíduos (variância não constante).
3. Não linearidade dos coeficientes da variável "ano de última reforma".

Por esses motivos, será aplicada uma regressão robusta, em detrimento da regressão linear.

A regressão robusta pode ser aplicada em variáveis explicativas que não possuem:

1. Normalidade dos resíduos
2. Homocedasticidade
3. Suficiente linearidade para aplicação da regressão linear

Dessa forma, será aplicada a regressão robusta de Huber, que é um exemplo de algoritmo de regressão robusto que atribui menos peso às observações identificadas como outliers.

In [10]:
```
import statsmodels.api as sm
from statsmodels.robust.robust_linear_model import RLM


X1 = sm.add_constant(X1)  # Adiciona o intercepto

# Regressão robusta com função de perda Huber
modelo_robusto = sm.RLM(y1, X1, M=sm.robust.norms.HuberT())
resultado = modelo_robusto.fit()

print(resultado.summary())
print()


modelo_robusto2 = sm.RLM(y1, X2, M=sm.robust.norms.HuberT())
resultado2 = modelo_robusto2.fit()

print(resultado2.summary())
print()
```

```
                    Robust linear Model Regression Results
==============================================================================
Dep. Variable:              SalePrice   No. Observations:                 2930
Model:                            RLM   Df Residuals:                     2928
Method:                          IRLS   Df Model:                            1
Norm:                          HuberT
Scale Est.:                       mad
Cov Type:                          H1
Date:                Thu, 10 Jul 2025
Time:                        17:53:42
No. Iterations:                     4
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const        1.526e+04   2549.631      5.984      0.000    1.03e+04    2.03e+04
Gr Liv Area   108.3161      1.611     67.232      0.000     105.158     111.474
==============================================================================

If the model instance has been used for another fit with different fit parameters,
then the fit options might not be the correct ones anymore .
```

```
                    Robust linear Model Regression Results
==============================================================================
Dep. Variable:              SalePrice   No. Observations:                 2930
Model:                            RLM   Df Residuals:                     2929
Method:                          IRLS   Df Model:                            0
Norm:                          HuberT
Scale Est.:                       mad
Cov Type:                          H1
Date:                Thu, 10 Jul 2025
Time:                        17:53:42
No. Iterations:                     5
==============================================================================
                    coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Year Remod/Add    85.9863      0.593    144.898      0.000      84.823      87.149
==============================================================================

If the model instance has been used for another fit with different fit parameters,
then the fit options might not be the correct ones anymore .
```

A Aplicação da regressão robusta com função de perda de Hebuber permitiu obter os seguintes coeficientes:

Variável	β_1 (intercepto)	β_2 (coef. angular)
Gr Liv Area	15.260	108.3161
Year Remod/Add	0	85.9863

Questão C - Avaliação do modelo

Para avaliação do modelo de regressão robusta aplicado, deve-se evitar utilizar o R^2, pois ele é sensível a outliers (o que pode ser observado nos dados) e também pela não normalidade dos resíduos. Portanto, o uso de R^2 pode indicar resultados equivocados.

Considerando-se o contexto, a métrica de avaliação mais adequada para uma regressão robusta com outliers e sem normalidade dos resíduos é a MAE (Mean Absolute Error), que mede o erro médio absoluto entre os valores preditos e os reais.

In [11]:
```
import statsmodels.api as sm
from statsmodels.robust.robust_linear_model import RLM
from sklearn.metrics import mean_absolute_error

# Adiciona o intercepto
X1 = sm.add_constant(X1)
X2 = sm.add_constant(X2)

# Modelo 1 - Gr Liv Area
modelo_robusto = sm.RLM(y1, X1, M=sm.robust.norms.HuberT())
resultado = modelo_robusto.fit()

# Previsões
y_pred1 = resultado.predict(X1)

# MAE
mae1 = mean_absolute_error(y1, y_pred1)
print(f"MAE (Gr Liv Area): {mae1:.2f}")

# Modelo 2 - Year Remod/Add
modelo_robusto2 = sm.RLM(y1, X2, M=sm.robust.norms.HuberT())
resultado2 = modelo_robusto2.fit()

# Previsões
y_pred2 = resultado2.predict(X2)

# MAE
mae2 = mean_absolute_error(y1, y_pred2)
print(f"MAE (Year Remod/Add): {mae2:.2f}")
```

```
MAE (Gr Liv Area): 38389.66
MAE (Year Remod/Add): 46037.75
```

Os valores acima obtidos mostram que o erro do modelo quando prevê o valor da casa de acordo com a variável explicativa área construída (Gr Liv Area) é de US 38.389,66. Enquanto que o erro utilizando-se a variável explicativa ano de última reforma/ampliação (Year Remod/add) é de US 46.037,75.

Por esse motivo, é possível afirmar que a variável explicativa Gr Liv Area é mais adequada para realizar previsões do valor de venda da casa.

É possível ainda calcular outra métrica que avalia o modelo, a MedAE, que calcula o erro a partir da mediana dos valores, e portanto não é afetada por valores extremos (outliers).

In [12]:
```
from sklearn.metrics import median_absolute_error

# Cálculo do MedAE reutilizando y_pred1 e y_pred2
medae1 = median_absolute_error(y1, y_pred1)
medae2 = median_absolute_error(y1, y_pred2)

print(f"MedAE (Gr Liv Area): {medae1:.2f}")
print(f"MedAE (Year Remod/Add): {medae2:.2f}")
```

```
MedAE (Gr Liv Area): 25994.09
MedAE (Year Remod/Add): 31580.46
```

O MedAE indica o erro do modelo entre o cálculo do valor predito e o valor real, considerando-se diferentes variáveis explicativas. Em outras palavras, é o valor pelo qual, em média, as previsões diferem dos dados observados, com a vantagem de ser mais robusto a outliers do que o MAE.

O modelo que usa área construída (Gr Liv Area) como variável explicativa erra em torno de US 25.994,09 no valor do imóvel, na mediana dos casos.

O modelo que usa ano da última reforma (Year Remod/Add) erra em torno de US 31.580,46, também na mediana dos casos.

Dessa forma, também por esse método de avaliação do erro, a variável explicativa Gr Liv Area se mostra mais adequada para prever o valor do imóvel.

O valor dos erros foi calculado de forma absoluta, o que torna difícil a avaliação do desempenho do modelo. Portanto, será calculado também em termos percentuais: o MAPE e o mdAPE

In [13]:
```
# Erros percentuais absolutos
erro_percentual1 = np.abs((y1 - y_pred1) / y1) * 100
erro_percentual2 = np.abs((y1 - y_pred2) / y1) * 100

# MAPE
mape1 = np.mean(erro_percentual1)
mape2 = np.mean(erro_percentual2)

# MdAPE
mdape1 = np.median(erro_percentual1)
mdape2 = np.median(erro_percentual2)

print(f"MAPE (Gr Liv Area): {mape1:.2f}%")
print(f"MdAPE (Gr Liv Area): {mdape1:.2f}%")
print()
print(f"MAPE (Year Remod/Add): {mape2:.2f}%")
print(f"MdAPE (Year Remod/Add): {mdape2:.2f}%")
```

```
MAPE (Gr Liv Area): 23.05%
MdAPE (Gr Liv Area): 15.53%

MAPE (Year Remod/Add): 26.79%
MdAPE (Year Remod/Add): 19.35%
```

A diferença no valor dos dois indicadores para cada variável indica a forte presença de outliers. Como o MdAPE calcula o erro percentual com base na mediana, a influência dos outliers é mitigada, reduzindo bastante o erro.

O mesmo ocorre para a variável explicativa Year Remod/Add.

Ao se analisar cada variável explicativa e seus resultados para os dois indicadores, percebe-se, da mesma forma como em termos absolutos, que em termos percentuais a Gr Liv Area se mostra mais adequada e com menos erro instrínseco para a previsão do valor do imóvel.

Questão D - Interação entre características

Será avaliada a interação entre as duas características Gr Liv Area e Year Remod/add para avaliar se a interação é estatisticamente significativa.

O termo de interação representa o efeito conjunto das duas variáveis explicativas sobre a variável dependente. No caso desse exercício, por exemplo, a presença de interação entre as variáveis representaria uma alteração na influência da área construída no valor do imóvel, a depender do ano de última reforma/ampliação.

Para captar a interação entre as características, será criada uma nova coluna, que receberá o produto das duas variáveis explicativas.

In [14]:
```
# df_clean = df[["Gr Liv Area", "SalePrice"]].dropna()
# df_clean2 = df[["Year Remod/Add", "SalePrice"]].dropna()

# # Variáveis
# X1 = df_clean[["Gr Liv Area"]] #variável explicativa
# y1 = df_clean["SalePrice"] # variável dependente

# X2 = df_clean2[["Year Remod/Add"]] #variável explicativa


df['interacao'] = (df['Gr Liv Area'] * df['Year Remod/Add']).dropna()


# Define X e y
X = df[['Gr Liv Area', 'Year Remod/Add', 'interacao']]
X = sm.add_constant(X)
y = df['SalePrice']

modelo_int = sm.RLM(y, X, M=sm.robust.norms.HuberT())

resultado_int = modelo_int.fit()
print(resultado_int.summary())
```

```
                    Robust linear Model Regression Results
==============================================================================
Dep. Variable:              SalePrice   No. Observations:                 2930
Model:                            RLM   Df Residuals:                     2926
Method:                          IRLS   Df Model:                            3
Norm:                          HuberT
Scale Est.:                       mad
Cov Type:                          H1
Date:                Thu, 10 Jul 2025
Time:                        17:53:42
No. Iterations:                     5
==============================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.274e+06   2.08e+05     10.918      0.000    1.87e+06    2.68e+06
Gr Liv Area   -3223.0925    141.468    -22.783      0.000   -3500.364   -2945.821
Year Remod/Add -1126.1247    104.980    -10.727      0.000   -1331.883    -920.367
interacao          1.6692      0.071     23.463      0.000       1.530       1.809
==============================================================================

If the model instance has been used for another fit with different fit parameters,
then the fit options might not be the correct ones anymore .
```

Admite-se a hipótese nula H0 para esse caso como sendo:

H0: Não há interação significativa entre as variáveis, ou seja, o coeficiente da "interação" é zero.

E hipótese alternativa: H1: há interação significativa entre as variáveis.

a linha interação do sumário mostrou um p valor baixo, o que indica a rejeição da hipótese nula, ou seja, eu rejeito que não há interação significativa. Portanto, há interação significativa entre as variáveis. Portanto, o efeito da área construída no preço varia conforme o ano da reforma/ampliação (ou vice-versa). O código a seguir mostrará essa interação na prática.

In [15]:
```python
import pandas as pd
import statsmodels.api as sm



# Coeficientes
b0 = resultado_int.params["const"]
b1 = resultado_int.params["Gr Liv Area"]
b2 = resultado_int.params["Year Remod/Add"]
b3 = resultado_int.params["interacao"]

# Valores simulados para GrLivArea
areas = [1000, 1500, 2000]

# Dois cenários de reforma
ano_antigo = 1950
ano_recente = 2010

print("Impacto no SalePrice com reformas antigas (1950):")
precos_antigos = []
for area in areas:
```

```
    preco = b0 + b1 * area + b2 * ano_antigo + b3 * (area * ano_antigo)
    precos_antigos.append(preco)
    print(f"Área: {area} - Preço estimado: {preco:,.2f}")

print("\nImpacto no SalePrice com reformas recentes (2010):")
for i, area in enumerate(areas):
    preco_recente = b0 + b1 * area + b2 * ano_recente + b3 * (area * ano_recente)
    preco_antigo = precos_antigos[i]
    variacao_percentual = ((preco_recente - preco_antigo) / preco_antigo) * 100
    print(f"Área: {area} - Preço estimado: {preco_recente:,.2f} "
          f"(+{variacao_percentual:.2f}% em relação a 1950)")
```

```
Impacto no SalePrice com reformas antigas (1950):
Área: 1000 - Preço estimado: 110,094.62
Área: 1500 - Preço estimado: 125,974.53
Área: 2000 - Preço estimado: 141,854.44

Impacto no SalePrice com reformas recentes (2010):
Área: 1000 - Preço estimado: 142,676.44 (+29.59% em relação a 1950)
Área: 1500 - Preço estimado: 208,631.00 (+65.61% em relação a 1950)
Área: 2000 - Preço estimado: 274,585.56 (+93.57% em relação a 1950)
```

Isso indica que, quando o ano da reforma é mais recente, o aumento na área construída tem um impacto muito maior no preço da casa.