In [1]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats
```

In [2]:
```
df_clean = pd.read_csv('ALL_STOCKS_MERGED_2.csv', parse_dates=['Date'])

df_clean = df_clean.drop_duplicates(subset=['Stock','Date'])
df_clean = df_clean.sort_values(['Stock','Date'])

# ffill per saham untuk OHLC (lebih aman)
for c in ['Open','High','Low','Close']:
    df_clean[c] = df_clean.groupby('Stock')[c].ffill()

# volume: kalau memang mau, tetap 0; tapi sadar konsekuensinya
df_clean['Volume'] = df_clean['Volume'].fillna(0)

df_clean = df_clean.dropna(subset=['Stock','Open','High','Low','Close'])
df_clean = df_clean.set_index('Date')

def resample_weekly_per_stock(df):
    df_resampled = (
        df.groupby('Stock')
          .resample('W-FRI')
          .agg({
              'Close': 'last',
              'Open':  'first',
              'High':  'max',
              'Low':   'min',
              'Volume':'sum'
          })
          .reset_index()
    )

    # eksplisit: jangan ada forward fill otomatis di pct_change
    df_resampled['return'] = (
        df_resampled
```

```
        .groupby('Stock')['Close']
        .pct_change(fill_method=None)
    )

    return df_resampled

df_weekly = resample_weekly_per_stock(df_clean).dropna(subset=['return'])
df_weekly = df_weekly[df_weekly['Stock'] != 'WSKT']
df_clean = df_weekly
print(f"Data setelah pembersihan: {df_weekly.isnull().sum()}")
df_weekly.head()
```

```
Data setelah pembersihan: Stock     0
Date      0
Close     0
Open      0
High      0
Low       0
Volume    0
return    0
dtype: int64
```

Out[2]:

	Stock	Date	Close	Open	High	Low	Volume	return
1	AALI	2014-01-10 00:00:00+07:00	14349.150391	15626.124588	15693.335662	14181.131010	6982727	-0.089552
2	AALI	2014-01-17 00:00:00+07:00	14029.906250	14332.346951	14651.589798	13878.684396	11189469	-0.022248
3	AALI	2014-01-24 00:00:00+07:00	15290.073242	14097.119811	15491.711148	14097.119811	12005338	0.089820
4	AALI	2014-01-31 00:00:00+07:00	14433.161133	14853.219393	14853.219393	14197.929047	6281672	-0.056044
5	AALI	2014-02-07 00:00:00+07:00	14886.824219	14433.161983	14987.637983	14281.941217	7120208	0.031432

In [3]:

```
def create_past_return_features(df, n_lags=15):
    for lag in range(1, n_lags + 1):
        df[f'past_return_t-{lag}'] = df.groupby('Stock')['return'].shift(lag)
    return df

df_features = create_past_return_features(df_weekly)

df_features = df_features.dropna(
```

```
    subset=[f'past_return_t-{lag}' for lag in range(1, 16)]
)

features_list = [f'past_return_t-{lag}' for lag in range(1, 16)]
target_col = 'return'

# Menampilkan beberapa data setelah penambahan fitur
df_features.head()
```

Out[3]:

	Stock	Date	Close	Open	High	Low	Volume	return	past_return_t-1	past_
16	AALI	2014-04-25 00:00:00+07:00	19020.185547	19154.604480	19809.891309	18448.908358	6268344	-0.001764	0.023465	
17	AALI	2014-05-02 00:00:00+07:00	19776.287109	19070.594682	19809.891695	19070.594682	4869699	0.039753	-0.001764	
18	AALI	2014-05-09 00:00:00+07:00	19641.869141	19490.647405	20061.926196	19356.230844	2768741	-0.006797	0.039753	-
19	AALI	2014-05-16 00:00:00+07:00	19713.304688	19951.429474	19951.429474	19049.956767	5836273	0.003637	-0.006797	
20	AALI	2014-05-23 00:00:00+07:00	18369.605469	19815.361328	19815.361328	18097.461987	10790561	-0.068162	0.003637	-

5 rows × 23 columns

In [4]:

```
print("\n=== BAGIAN 3: TRAINING MODEL (SEMUA SAHAM) ===")

results = []
prediksi_per_saham = {}

stock_list = df_features['Stock'].unique()
total_stocks = len(stock_list)

for i, stock in enumerate(stock_list):
    print(f"[{i+1}/{total_stocks}] Memproses {stock}...", end=" ")

    # ====================
```

```
    # 1. Split Data
    # ======================
    df_s = df_features[df_features['Stock'] == stock]
    max_date = df_s['Date'].max()
    cutoff_date = max_date - pd.DateOffset(years=3)

    train = df_s[df_s['Date'] <= cutoff_date]
    test  = df_s[df_s['Date'] > cutoff_date]

    if len(train) < 50 or len(test) < 10:
        print("SKIP (Data kurang)")
        continue

    # ======================
    # 2. Train Random Forest
    # ======================
    rf = RandomForestRegressor(
        n_estimators=500,
        max_depth=20,
        random_state=42
    )

    rf.fit(train[features_list], train[target_col])

    # ======================
    # 3. PREDIKSI TRAIN
    # ======================
    train_preds = rf.predict(train[features_list])

    df_train = train.copy()
    df_train['Prediksi_Return'] = train_preds
    df_train['Actual_Return'] = train[target_col]
    df_train['Set'] = 'Train'

    train_rmse = np.sqrt(mean_squared_error(train[target_col], train_preds))
    train_mae  = mean_absolute_error(train[target_col], train_preds)

    # ======================
    # 4. PREDIKSI TEST
    # ======================
    test_preds = rf.predict(test[features_list])
```

```
df_test = test.copy()
df_test['Prediksi_Return'] = test_preds
df_test['Actual_Return'] = test[target_col]
df_test['Set'] = 'Test'

test_rmse = np.sqrt(mean_squared_error(test[target_col], test_preds))
test_mae  = mean_absolute_error(test[target_col], test_preds)

# ======================
# 5. GABUNGKAN TRAIN + TEST
# ======================
df_all = pd.concat([df_train, df_test])

prediksi_per_saham[stock] = df_all[
    ['Date', 'Stock', 'Close',
     'Actual_Return', 'Prediksi_Return', 'Set']
]

# ======================
# 6. HIT RATIO (TEST SAJA)
# ======================
actual_dir = np.sign(test[target_col])
pred_dir   = np.sign(test_preds)

hr = (actual_dir == pred_dir).mean()

mask_plus  = (actual_dir > 0)
hr_plus  = (pred_dir[mask_plus] > 0).mean() if mask_plus.sum() > 0 else np.nan

mask_minus = (actual_dir < 0)
hr_minus = (pred_dir[mask_minus] < 0).mean() if mask_minus.sum() > 0 else np.nan

# ======================
# 7. PRINT RINGKAS
# ======================
print(
    f"{stock} | "
    f"TRAIN | RMSE: {train_rmse:.4f} | MAE: {train_mae:.4f} || "
    f"TEST  | RMSE: {test_rmse:.4f} | MAE: {test_mae:.4f} | "
    f"HR: {hr:.1%}"
```

```
    )

    # ====================
    # 8. SIMPAN HASIL
    # ====================
    results.append({
        'Stock': stock,
        'Train_MAE': train_mae,
        'Train_RMSE': train_rmse,
        'Test_MAE': test_mae,
        'Test_RMSE': test_rmse,
        'HR': hr,
        'HR+': hr_plus,
        'HR-': hr_minus
    })

print("Training selesai. Lanjutkan ke Bagian 4.")
```

```
=== BAGIAN 3: TRAINING MODEL (SEMUA SAHAM) ===
AALI | TRAIN | RMSE: 0.0228 | MAE: 0.0158 || TEST  | RMSE: 0.0338 | MAE: 0.0226 | HR: 51.6%
ACES | TRAIN | RMSE: 0.0195 | MAE: 0.0144 || TEST  | RMSE: 0.0609 | MAE: 0.0459 | HR: 48.4%
ADHI | TRAIN | RMSE: 0.0280 | MAE: 0.0190 || TEST  | RMSE: 0.0634 | MAE: 0.0474 | HR: 49.0%
ADRO | TRAIN | RMSE: 0.0260 | MAE: 0.0192 || TEST  | RMSE: 0.0560 | MAE: 0.0430 | HR: 48.4%
AKRA | TRAIN | RMSE: 0.0200 | MAE: 0.0144 || TEST  | RMSE: 0.0482 | MAE: 0.0365 | HR: 48.4%
AMRT | TRAIN | RMSE: 0.0185 | MAE: 0.0127 || TEST  | RMSE: 0.0470 | MAE: 0.0335 | HR: 52.3%
ANTM | TRAIN | RMSE: 0.0280 | MAE: 0.0194 || TEST  | RMSE: 0.0539 | MAE: 0.0424 | HR: 52.9%
ASII | TRAIN | RMSE: 0.0170 | MAE: 0.0118 || TEST  | RMSE: 0.0327 | MAE: 0.0252 | HR: 44.4%
BBCA | TRAIN | RMSE: 0.0124 | MAE: 0.0084 || TEST  | RMSE: 0.0214 | MAE: 0.0168 | HR: 47.1%
BBNI | TRAIN | RMSE: 0.0194 | MAE: 0.0135 || TEST  | RMSE: 0.0358 | MAE: 0.0275 | HR: 53.6%
BBRI | TRAIN | RMSE: 0.0176 | MAE: 0.0122 || TEST  | RMSE: 0.0300 | MAE: 0.0241 | HR: 49.0%
BBTN | TRAIN | RMSE: 0.0231 | MAE: 0.0161 || TEST  | RMSE: 0.0408 | MAE: 0.0296 | HR: 49.7%
BDMN | TRAIN | RMSE: 0.0243 | MAE: 0.0169 || TEST  | RMSE: 0.0317 | MAE: 0.0228 | HR: 41.2%
BMRI | TRAIN | RMSE: 0.0166 | MAE: 0.0118 || TEST  | RMSE: 0.0339 | MAE: 0.0254 | HR: 50.3%
BSDE | TRAIN | RMSE: 0.0200 | MAE: 0.0146 || TEST  | RMSE: 0.0365 | MAE: 0.0273 | HR: 41.8%
CTRA | TRAIN | RMSE: 0.0261 | MAE: 0.0188 || TEST  | RMSE: 0.0430 | MAE: 0.0316 | HR: 48.4%
EXCL | TRAIN | RMSE: 0.0220 | MAE: 0.0161 || TEST  | RMSE: 0.0467 | MAE: 0.0350 | HR: 44.4%
GGRM | TRAIN | RMSE: 0.0168 | MAE: 0.0124 || TEST  | RMSE: 0.0457 | MAE: 0.0298 | HR: 53.6%
HMSP | TRAIN | RMSE: 0.0165 | MAE: 0.0119 || TEST  | RMSE: 0.0395 | MAE: 0.0283 | HR: 50.3%
ICBP | TRAIN | RMSE: 0.0138 | MAE: 0.0098 || TEST  | RMSE: 0.0311 | MAE: 0.0238 | HR: 54.9%
INDF | TRAIN | RMSE: 0.0147 | MAE: 0.0103 || TEST  | RMSE: 0.0240 | MAE: 0.0186 | HR: 50.3%
INDY | TRAIN | RMSE: 0.0384 | MAE: 0.0265 || TEST  | RMSE: 0.0784 | MAE: 0.0545 | HR: 51.0%
INTP | TRAIN | RMSE: 0.0208 | MAE: 0.0158 || TEST  | RMSE: 0.0386 | MAE: 0.0272 | HR: 55.6%
ITMG | TRAIN | RMSE: 0.0257 | MAE: 0.0196 || TEST  | RMSE: 0.0455 | MAE: 0.0348 | HR: 52.3%
JSMR | TRAIN | RMSE: 0.0180 | MAE: 0.0128 || TEST  | RMSE: 0.0443 | MAE: 0.0316 | HR: 52.3%
KLBF | TRAIN | RMSE: 0.0159 | MAE: 0.0113 || TEST  | RMSE: 0.0337 | MAE: 0.0251 | HR: 54.9%
MEDC | TRAIN | RMSE: 0.0320 | MAE: 0.0221 || TEST  | RMSE: 0.0685 | MAE: 0.0495 | HR: 46.4%
MNCN | TRAIN | RMSE: 0.0243 | MAE: 0.0182 || TEST  | RMSE: 0.0476 | MAE: 0.0361 | HR: 44.4%
PGAS | TRAIN | RMSE: 0.0248 | MAE: 0.0180 || TEST  | RMSE: 0.0435 | MAE: 0.0327 | HR: 44.4%
PTBA | TRAIN | RMSE: 0.0218 | MAE: 0.0166 || TEST  | RMSE: 0.0469 | MAE: 0.0351 | HR: 41.2%
PTPP | TRAIN | RMSE: 0.0250 | MAE: 0.0184 || TEST  | RMSE: 0.0727 | MAE: 0.0501 | HR: 50.3%
SCMA | TRAIN | RMSE: 0.0210 | MAE: 0.0158 || TEST  | RMSE: 0.0632 | MAE: 0.0470 | HR: 46.4%
SIDO | TRAIN | RMSE: 0.0155 | MAE: 0.0108 || TEST  | RMSE: 0.0419 | MAE: 0.0304 | HR: 36.6%
SMGR | TRAIN | RMSE: 0.0199 | MAE: 0.0146 || TEST  | RMSE: 0.0452 | MAE: 0.0350 | HR: 50.3%
SMRA | TRAIN | RMSE: 0.0242 | MAE: 0.0182 || TEST  | RMSE: 0.0506 | MAE: 0.0385 | HR: 53.6%
TBIG | TRAIN | RMSE: 0.0211 | MAE: 0.0152 || TEST  | RMSE: 0.0357 | MAE: 0.0263 | HR: 47.7%
TLKM | TRAIN | RMSE: 0.0132 | MAE: 0.0099 || TEST  | RMSE: 0.0353 | MAE: 0.0261 | HR: 51.0%
TPIA | TRAIN | RMSE: 0.0191 | MAE: 0.0135 || TEST  | RMSE: 0.0905 | MAE: 0.0535 | HR: 49.7%
UNTR | TRAIN | RMSE: 0.0182 | MAE: 0.0138 || TEST  | RMSE: 0.0432 | MAE: 0.0324 | HR: 56.2%
```

```
UNVR | TRAIN | RMSE: 0.0142 | MAE: 0.0102 || TEST  | RMSE: 0.0467 | MAE: 0.0336 | HR: 55.6%
Training selesai. Lanjutkan ke Bagian 4.
```

In [5]:
```python
print("\n=== BAGIAN 4: HASIL EVALUASI ===")

# ======================
# 1. DataFrame hasil
# ======================
res_df = pd.DataFrame(results)

# ======================
# 2. Top 5 Saham Terbaik (berdasarkan TEST RMSE)
# ======================
print("\nTop 5 Saham dengan TEST RMSE Terkecil:")
print(
    res_df
    .sort_values('Test_RMSE')
    .head(5)[
        ['Stock', 'Train_RMSE', 'Test_RMSE',
         'Train_MAE', 'Test_MAE',
         'HR']
    ]
)

# ======================
# 3. Rata-rata Performansi Seluruh Saham
# ======================
print("\nRata-rata Performansi Seluruh Saham:")
print(
    res_df[
        ['Train_RMSE', 'Test_RMSE',
         'Train_MAE', 'Test_MAE',
         'HR']
    ].mean()
)

# ======================
# 4. Median Performansi (opsional tapi bagus)
# ======================
print("\nMedian Performansi Seluruh Saham:")
print(
```

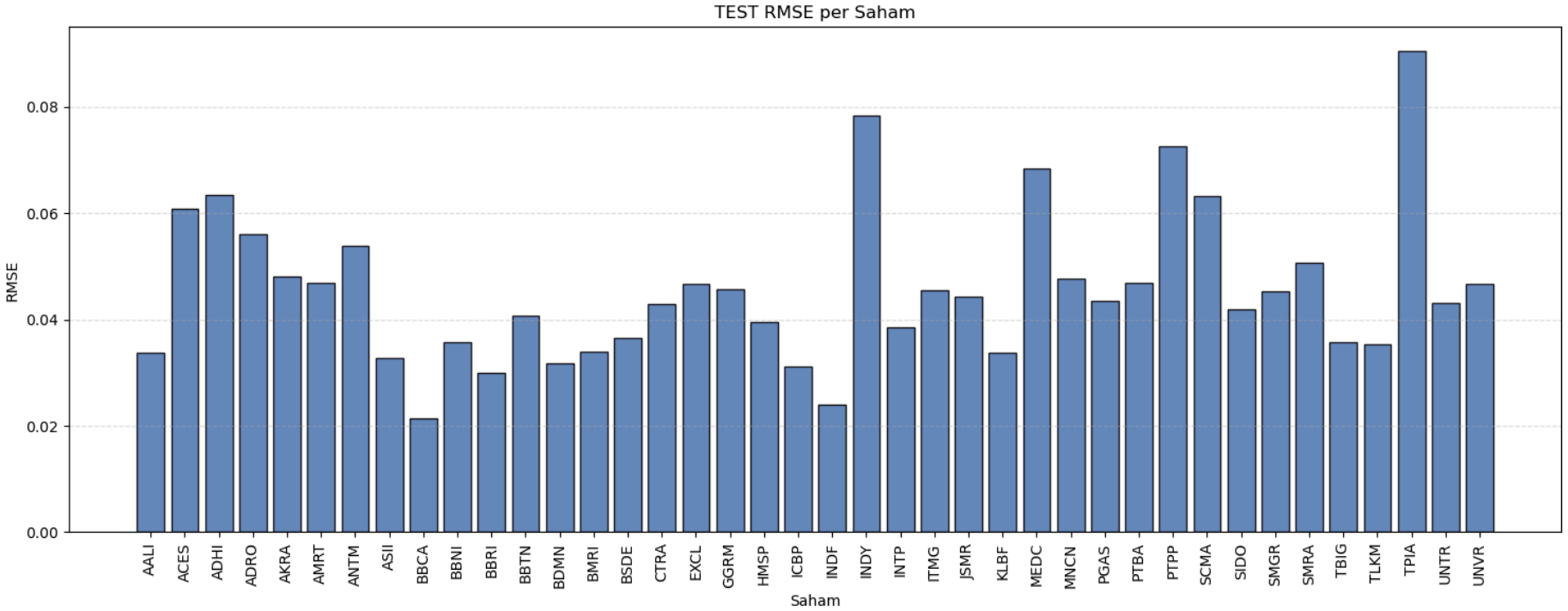
```
    res_df[
        ['Train_RMSE', 'Test_RMSE',
         'Train_MAE', 'Test_MAE',
         'HR']
    ].median()
)

# =====================
# 5. Gabungkan Semua Prediksi Saham
# =====================
df_all_predictions = pd.concat(
    prediksi_per_saham.values(),
    ignore_index=True
).sort_values(
    by=['Stock', 'Date']
)

# (Opsional) Cek jumlah Train vs Test
print("\nJumlah data Train vs Test:")
print(df_all_predictions['Set'].value_counts())

# =====================
# 6. Simpan ke CSV
# =====================
df_all_predictions.to_csv(
    'Hasil_Prediksi_Saham_15_2.csv',
    index=False
)

print("\nFile 'Hasil_Prediksi_Saham_15_2.csv' berhasil disimpan.")
```

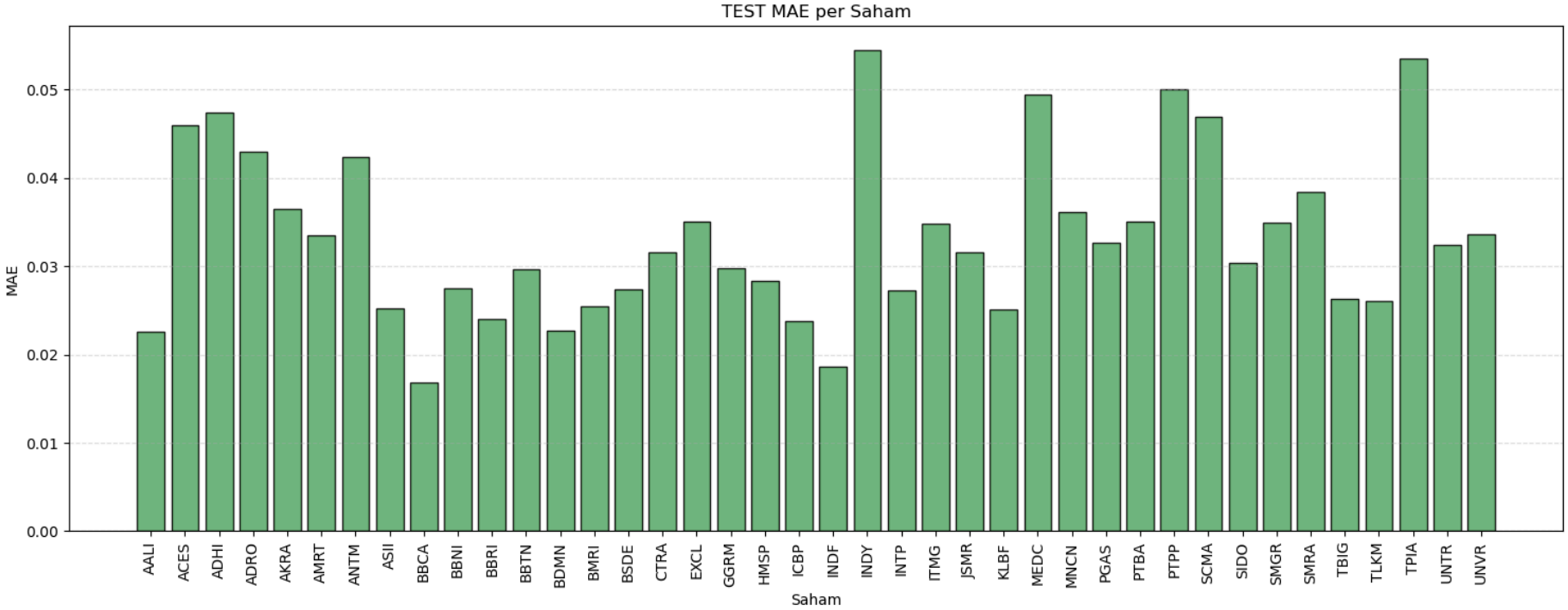
```
=== BAGIAN 4: HASIL EVALUASI ===

Top 5 Saham dengan TEST RMSE Terkecil:
   Stock  Train_RMSE  Test_RMSE  Train_MAE  Test_MAE        HR
8   BBCA    0.012390   0.021371   0.008397  0.016803  0.470588
20  INDF    0.014738   0.023954   0.010289  0.018619  0.503268
10  BBRI    0.017588   0.029953   0.012221  0.024085  0.490196
19  ICBP    0.013791   0.031144   0.009849  0.023803  0.549020
12  BDMN    0.024308   0.031687   0.016944  0.022764  0.411765

Rata-rata Performansi Seluruh Saham:
Train_RMSE    0.020912
Test_RMSE     0.045704
Train_MAE     0.015026
Test_MAE      0.033410
HR            0.492484
dtype: float64

Median Performansi Seluruh Saham:
Train_RMSE    0.020006
Test_RMSE     0.043919
Train_MAE     0.014620
Test_MAE      0.032007
HR            0.500000
dtype: float64

Jumlah data Train vs Test:
Set
Train    15920
Test      6120
Name: count, dtype: int64

File 'Hasil_Prediksi_Saham_15_2.csv' berhasil disimpan.
```

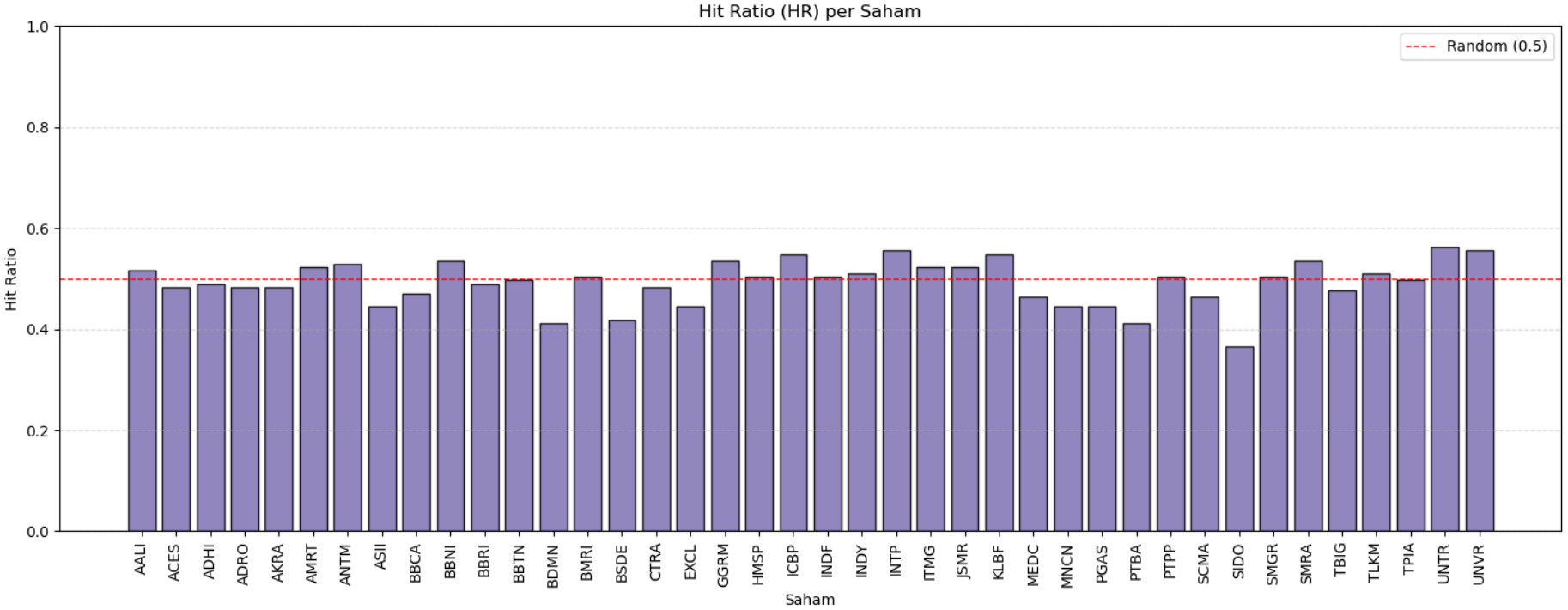
In [6]:
```python
def plot_bar_rmse_natural(res_df, use_test=True):
    metric = 'Test_RMSE' if use_test else 'Train_RMSE'
    title  = 'TEST RMSE per Saham' if use_test else 'TRAIN RMSE per Saham'
    color  = '#4C72B0'  # biru soft

    df_plot = res_df.copy()
```

```
    plt.figure(figsize=(15, 6))
    plt.bar(df_plot['Stock'], df_plot[metric],
            color=color, edgecolor='black', alpha=0.85)
    plt.xticks(rotation=90)
    plt.ylabel('RMSE')
    plt.xlabel('Saham')
    plt.title(title)
    plt.grid(axis='y', linestyle='--', alpha=0.4)
    plt.tight_layout()
    plt.show()

def plot_bar_mae_natural(res_df, use_test=True):
    metric = 'Test_MAE' if use_test else 'Train_MAE'
    title  = 'TEST MAE per Saham' if use_test else 'TRAIN MAE per Saham'
    color  = '#55A868'  # hijau soft

    df_plot = res_df.copy()

    plt.figure(figsize=(15, 6))
    plt.bar(df_plot['Stock'], df_plot[metric],
            color=color, edgecolor='black', alpha=0.85)
    plt.xticks(rotation=90)
    plt.ylabel('MAE')
    plt.xlabel('Saham')
    plt.title(title)
    plt.grid(axis='y', linestyle='--', alpha=0.4)
    plt.tight_layout()
    plt.show()
def plot_bar_hr_natural(res_df):
    df_plot = res_df.copy()
    color = '#8172B3'  # ungu soft

    plt.figure(figsize=(15, 6))
    plt.bar(df_plot['Stock'], df_plot['HR'],
            color=color, edgecolor='black', alpha=0.85)

    plt.axhline(0.5, color='red', linestyle='--', linewidth=1,
                label='Random (0.5)')

    plt.xticks(rotation=90)
    plt.ylabel('Hit Ratio')
```

```
    plt.xlabel('Saham')
    plt.title('Hit Ratio (HR) per Saham')
    plt.ylim(0, 1)
    plt.legend()
    plt.grid(axis='y', linestyle='--', alpha=0.4)
    plt.tight_layout()
    plt.show()

plot_bar_rmse_natural(res_df)
plot_bar_mae_natural(res_df)
plot_bar_hr_natural(res_df)
```

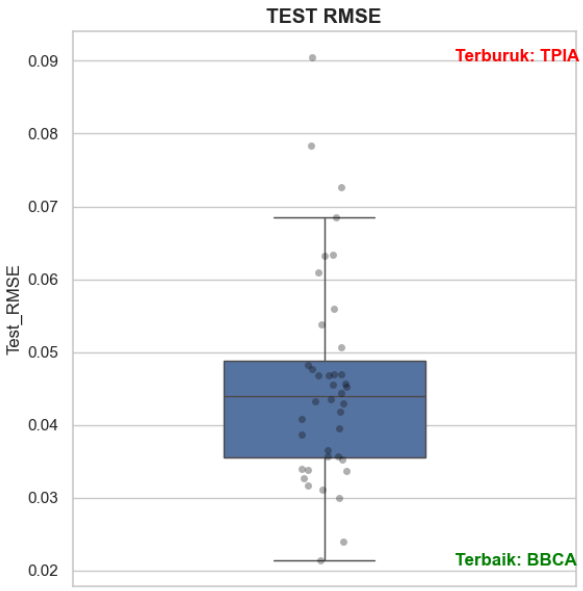


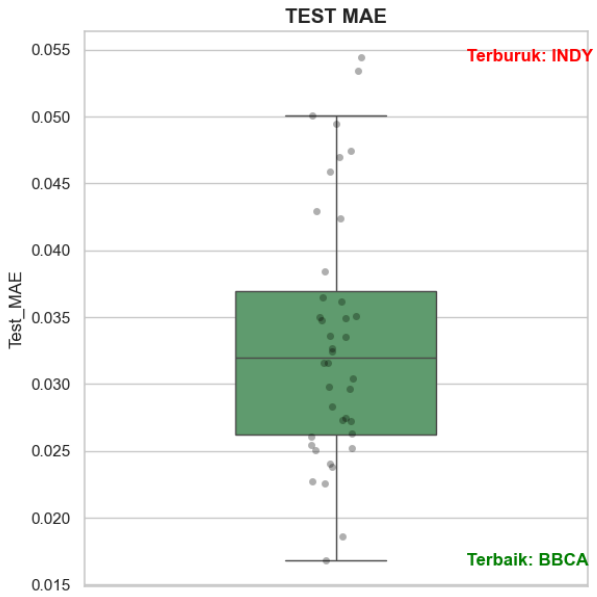

TEST MAE per Saham
MAE
0.00
0.01
0.02
0.03
0.04
0.05
AALI
ACES
ADHI
ADRO
AKRA
AMRT
ANTM
ASII
BBCA
BBNI
BBRI
BBTN
BDMN
BMRI
BSDE
CTRA
EXCL
GGRM
HMSP
ICBP
INDF
INDY
INTP
ITMG
JSMR
KLBF
MEDC
MNCN
PGAS
PTBA
PTPP
SCMA
SIDO
SMGR
SMRA
TBIG
TLKM
TPIA
UNTR
UNVR
Saham

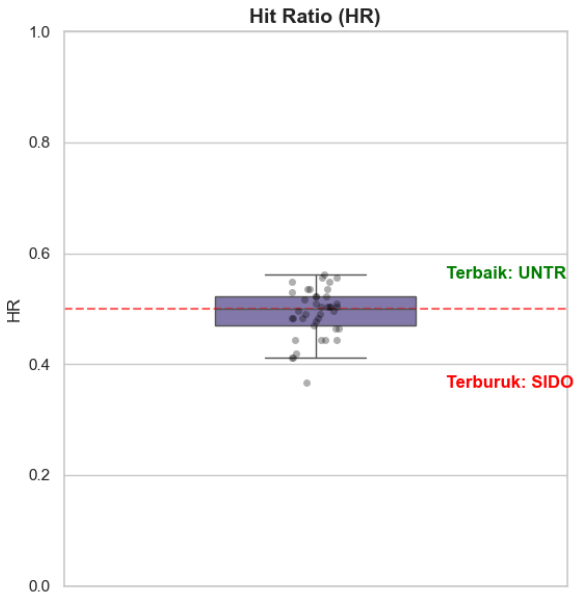


In [7]:
```
import matplotlib.pyplot as plt
import seaborn as sns

def plot_performance_with_labels(res_df):
    sns.set_theme(style="whitegrid")
    fig, axes = plt.subplots(1, 3, figsize=(18, 6))

    # Konfigurasi: (Nama Kolom, Judul, Index Grafik, Warna Asli Kamu, Target Nilai)
    config = [
        ('Test_RMSE', 'TEST RMSE', 0, '#4C72B0', 'low'),
        ('Test_MAE', 'TEST MAE', 1, '#55A868', 'low'),
        ('HR', 'Hit Ratio (HR)', 2, '#8172B3', 'high')
    ]

    for col, title, ax_idx, original_color, goal in config:
        ax = axes[ax_idx]

        # Plot Boxplot dengan warna asli kamu
```

```
        sns.boxplot(y=res_df[col], ax=ax, color=original_color, width=0.4, fliersize=0)
        # Tambahkan titik data agar terlihat sebaran sahamnya
        sns.stripplot(y=res_df[col], ax=ax, color='black', alpha=0.3, jitter=0.05)

        # Logika mencari terbaik dan terjelek
        best_val = res_df[col].min() if goal == 'low' else res_df[col].max()
        worst_val = res_df[col].max() if goal == 'low' else res_df[col].min()

        best_stock = res_df.loc[res_df[col] == best_val, 'Stock'].values[0]
        worst_stock = res_df.loc[res_df[col] == worst_val, 'Stock'].values[0]

        # Anotasi Teks
        ax.text(0.25, best_val, f' Terbaik: {best_stock}', color='green', fontweight='bold', va='center')
        ax.text(0.25, worst_val, f' Terburuk: {worst_stock}', color='red', fontweight='bold', va='center')

        ax.set_title(title, fontsize=14, fontweight='bold')
        if col == 'HR':
            ax.axhline(0.5, color='red', linestyle='--', alpha=0.6)
            ax.set_ylim(0, 1)

    plt.tight_layout()
    plt.show()

# Jalankan fungsi
plot_performance_with_labels(res_df)
```





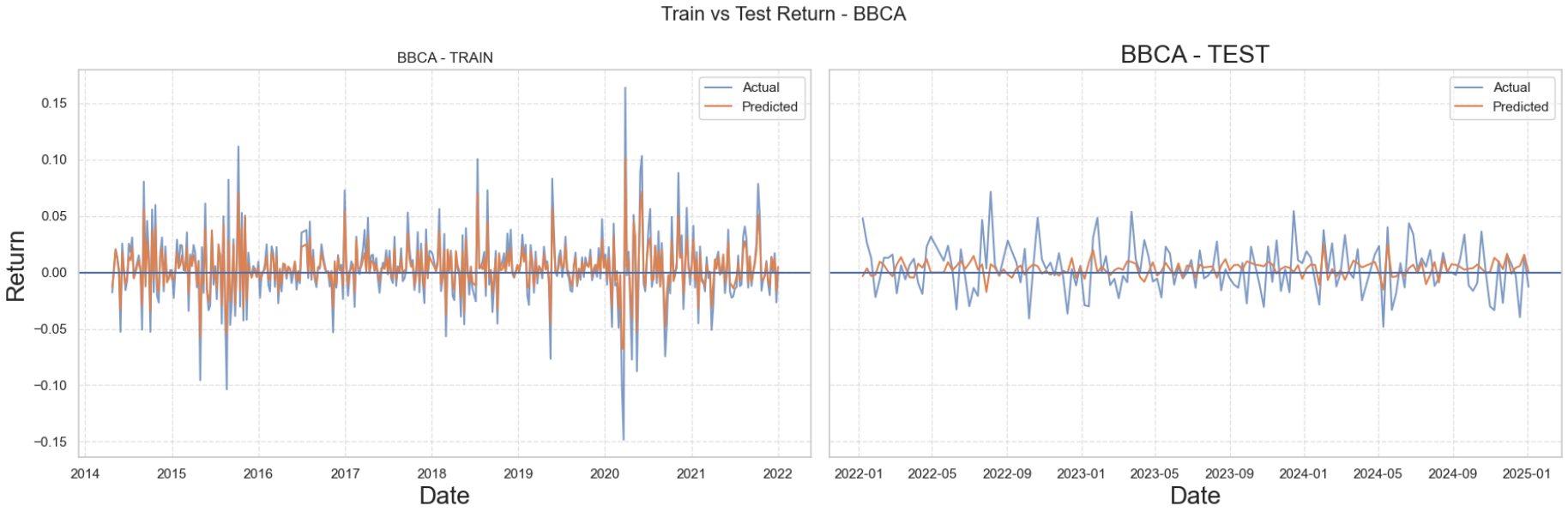


In [8]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import calendar  # untuk menentukan akhir bulan otomatis

# ==========================================
# 1. LOAD & PREPARE DATA
# ==========================================
dc = pd.read_csv("Hasil_Prediksi_Saham_15_2.csv")
dc["Date"] = pd.to_datetime(dc["Date"])
dc = dc.sort_values(["Stock", "Date"]).reset_index(drop=True)

# Simpan ke variabel resmi untuk plotting
dc_plot = dc.copy()

# ==========================================
# 2. FUNGSI PLOTTING UTAMA
# ==========================================
def plot_train_vs_test(stock, start_date=None, end_date=None):
    import matplotlib.pyplot as plt

    df = dc_plot[dc_plot["Stock"] == stock].copy()
```

```
    # Filter tanggal (opsional)
    if start_date and end_date:
        df = df[(df["Date"] >= start_date) & (df["Date"] <= end_date)]

    df_train = df[df["Set"] == "Train"]
    df_test  = df[df["Set"] == "Test"]

    if df_train.empty or df_test.empty:
        print(f"Data Train/Test tidak lengkap untuk {stock}")
        return

    fig, axes = plt.subplots(1, 2, figsize=(18, 6), sharey=True)

    # ===== KIRI: TRAIN =====
    axes[0].plot(df_train["Date"], df_train["Actual_Return"], label="Actual", alpha=0.7)
    axes[0].plot(df_train["Date"], df_train["Prediksi_Return"], label="Predicted")
    axes[0].axhline(0)
    axes[0].set_title(f"{stock} - TRAIN")
    axes[0].set_xlabel("Date", fontsize=20)
    axes[0].set_ylabel("Return", fontsize=20)
    axes[0].legend()
    axes[0].grid(True, linestyle="--", alpha=0.5)

    # ===== KANAN: TEST =====
    axes[1].plot(df_test["Date"], df_test["Actual_Return"], label="Actual", alpha=0.7)
    axes[1].plot(df_test["Date"], df_test["Prediksi_Return"], label="Predicted")
    axes[1].axhline(0)
    axes[1].set_title(f"{stock} - TEST", fontsize=20)
    axes[1].set_xlabel("Date",fontsize=20)
    axes[1].legend()
    axes[1].grid(True, linestyle="--", alpha=0.5)

    plt.suptitle(f"Train vs Test Return - {stock}", fontsize=16)
    plt.tight_layout()
    plt.show()

def plot_train_vs_test_all():
    stock_list = dc_plot['Stock'].unique()
    print(f"Menampilkan {len(stock_list)} saham...\n")
```

```
    for i, stock in enumerate(stock_list, 1):
        print(f"[{i}/{len(stock_list)}] Plotting {stock}")
        plot_train_vs_test(stock)

def plot_top10_train_vs_test(metric='Test_RMSE'):
    # Ambil Top 10 saham berdasarkan metric (default: Test_RMSE)
    top10 = (
        res_df
        .sort_values(metric)
        .head(10)['Stock']
        .tolist()
    )

    fig, axes = plt.subplots(
        nrows=10, ncols=2,
        figsize=(14, 22),
        sharey=True
    )

    for i, stock in enumerate(top10):
        df = dc_plot[dc_plot['Stock'] == stock]

        df_train = df[df['Set'] == 'Train']
        df_test  = df[df['Set'] == 'Test']

        # ===== TRAIN (KIRI) =====
        axes[i, 0].plot(df_train['Date'], df_train['Actual_Return'],
                        linewidth=0.8, label='Actual')
        axes[i, 0].plot(df_train['Date'], df_train['Prediksi_Return'],
                        linewidth=0.8, linestyle='--', label='Predicted')
        axes[i, 0].axhline(0, linewidth=0.5)
        axes[i, 0].set_title(f"{stock} - TRAIN", fontsize=9)
        axes[i, 0].grid(True, linestyle='--', alpha=0.3)

        # ===== TEST (KANAN) =====
        axes[i, 1].plot(df_test['Date'], df_test['Actual_Return'],
                        linewidth=0.8, label='Actual')
        axes[i, 1].plot(df_test['Date'], df_test['Prediksi_Return'],
                        linewidth=0.8, linestyle='--', label='Predicted')
        axes[i, 1].axhline(0, linewidth=0.5)
        axes[i, 1].set_title(f"{stock} - TEST", fontsize=9)
```

```
        axes[i, 1].grid(True, linestyle='--', alpha=0.3)

        if i == 0:
            axes[i, 0].legend(fontsize=8)
            axes[i, 1].legend(fontsize=8)

    plt.suptitle(
        'Visualisasi Train vs Test Return — 10 Saham dengan Error Terkecil',
        fontsize=16
    )
    plt.tight_layout(rect=[0, 0, 1, 0.97])
    plt.show()



plot_train_vs_test("BBCA")
#plot_train_vs_test_all()
#plot_top10_train_vs_test()
```



In [9]:
```
import pandas as pd
import numpy as np
```

```
# ==========================================
# 1. LOAD & SORTING DATA
# ==========================================
print("Loading data...")

dc = pd.read_csv("Hasil_Prediksi_Saham_15.csv")
dc["Date"] = pd.to_datetime(dc["Date"])

# MODIFIKASI: Sort berdasarkan Stock dulu, baru Date
# Ini akan membuat urutan: AALI (2014-2025), ACES (2014-2025), dst.
dc = dc.sort_values(["Stock", "Date"])

# ==========================================
# 2. MEMBUAT DC_TEST
# ==========================================
# Karena dc sudah disort per Stock, dc_test juga akan otomatis urut per Stock
dc_test = dc[dc["Set"] == "Test"].copy()

print("Data berhasil dimuat dan diurutkan.")
print("\nContoh Head Data Full (Lihat AALI mulai dari 2014):")
print(dc.head())

print("\nContoh Head Data Test (Lihat AALI mulai dari Test set):")
print(dc_test.head())
```

```
Loading data...
Data berhasil dimuat dan diurutkan.

Contoh Head Data Full (Lihat AALI mulai dari 2014):
                       Date Stock         Close  Actual_Return  \
0 2014-04-25 00:00:00+07:00  AALI  19020.185547      -0.001764
1 2014-05-02 00:00:00+07:00  AALI  19776.287109       0.039753
2 2014-05-09 00:00:00+07:00  AALI  19641.869141      -0.006797
3 2014-05-16 00:00:00+07:00  AALI  19713.304688       0.003637
4 2014-05-23 00:00:00+07:00  AALI  18369.605469      -0.068162

   Prediksi_Return    Set
0         0.000783  Train
1         0.018375  Train
2        -0.006368  Train
3        -0.002865  Train
4        -0.045491  Train

Contoh Head Data Test (Lihat AALI mulai dari Test set):
                         Date Stock        Close  Actual_Return  \
398 2022-01-07 00:00:00+07:00  AALI  8358.950195       0.047368
399 2022-01-14 00:00:00+07:00  AALI  8379.954102       0.002513
400 2022-01-21 00:00:00+07:00  AALI  8337.950195      -0.005012
401 2022-01-28 00:00:00+07:00  AALI  8127.925781      -0.025189
402 2022-02-04 00:00:00+07:00  AALI  8169.930664       0.005168

     Prediksi_Return   Set
398         0.053035  Test
399        -0.014048  Test
400        -0.011537  Test
401        -0.010247  Test
402        -0.003199  Test
```

In [10]:

```python
import pandas as pd
import numpy as np
from scipy.optimize import minimize

# ==========================================
# 1. LOAD DATA
# ==========================================
print("Loading data...")
```

```
dc = pd.read_csv("Hasil_Prediksi_Saham.csv")
dc["Date"] = pd.to_datetime(dc["Date"])
dc = dc.sort_values(["Stock", "Date"])

# Forecast error
dc_test = dc[dc["Set"] == "Test"].copy()
dc_test["Weekly_Error"] = dc_test["Actual_Return"] - dc_test["Prediksi_Return"]
dc_test["Rolling_Error_20"] = (dc_test.groupby("Stock")["Weekly_Error"].transform(lambda x: x.rolling(20).mean().sh

dc_clean = dc_test.dropna(subset=["Rolling_Error_20"]).copy()
dc_clean.head()
```

```
Loading data...
```

Out[10]:

	Date	Stock	Close	Actual_Return	Prediksi_Return	Set	Weekly_Error	Rolling_Error_20
413	2022-06-10 00:00:00+07:00	AALI	9500.416992	-0.073684	0.052092	Test	-0.125776	0.016411
414	2022-06-17 00:00:00+07:00	AALI	8679.926758	-0.086364	-0.003329	Test	-0.083035	0.009375
415	2022-06-24 00:00:00+07:00	AALI	8464.008789	-0.024876	-0.000513	Test	-0.024363	0.004609
416	2022-07-01 00:00:00+07:00	AALI	8334.457031	-0.015306	0.002503	Test	-0.017809	0.003008
417	2022-07-08 00:00:00+07:00	AALI	8269.681641	-0.007772	-0.012502	Test	0.004730	0.003277

In [11]:

```
import pandas as pd
import numpy as np
from scipy.optimize import minimize

# ============================================
# 1. LOAD DATA
# ============================================
print("Loading data...")

dc = pd.read_csv("Hasil_Prediksi_Saham_15_2.csv")
dc["Date"] = pd.to_datetime(dc["Date"])
dc = dc.sort_values(["Stock", "Date"])

dc_test = dc[dc["Set"] == "Test"].copy()
dc_test["Weekly_Error"] = dc_test["Actual_Return"] - dc_test["Prediksi_Return"]
```

```
dc_test["Rolling_Error_15"] = (
    dc_test.groupby("Stock")["Weekly_Error"]
    .transform(lambda x: x.rolling(15).mean().shift(1))
)
dc_clean = dc_test.dropna(subset=["Rolling_Error_15"]).copy()

dc_full = dc.copy()

stocks_list = sorted(dc_full["Stock"].unique())
n = len(stocks_list)

WINDOW_SIZE = 400
SCALE = 100000

print("Done.\n")

# ==========================================
# 2. TANGGAL BACKTEST
# ==========================================
pred_dates = np.sort(dc_clean["Date"].unique())

# ==========================================
# 3. 4 SKENARIO MVF (SEPADAN DENGAN MV)
# ==========================================
scenarios = {
    "MVF_S1_Equal":    (1/3, 1/3, 1/3),
    "MVF_S2_RiskUp":   (0.60, 0.20, 0.20),
    "MVF_S3_RiskMore": (0.70, 0.15, 0.15),
    "MVF_S4_VeryRisk": (0.80, 0.10, 0.10)
}

portfolio_return_mvf = {k: [] for k in scenarios}
log_ret_bench = []
dates_backtest = []

# ==========================================
# 4. BACKTEST (GAYA MV KLASIK)
# ==========================================
for today in pred_dates:

    hist_data = (
```

```
        dc_full[dc_full["Date"] < today]
        .sort_values("Date")
        .tail(WINDOW_SIZE * n)
    )

    if hist_data["Date"].nunique() < WINDOW_SIZE:
        continue

    pivot_train = (
        hist_data.pivot(index="Date", columns="Stock", values="Actual_Return")
        .reindex(columns=stocks_list)
        .fillna(0)
    )

    Sigma = pivot_train.cov().values

    today_pred = dc_clean[dc_clean["Date"] == today].set_index("Stock")
    r_hat = today_pred["Prediksi_Return"].reindex(stocks_list).fillna(0).values
    epsilon = today_pred["Rolling_Error_15"].reindex(stocks_list).fillna(0).values

    cons = ({'type': 'eq', 'fun': lambda w: np.sum(w) - 1})
    bnds = tuple((0.0, 1.0) for _ in range(n))
    init = np.ones(n) / n

    weights = {}

    for name, (a1, a2, a3) in scenarios.items():

        def obj_mvf(w):
            return (
                0.5 * a1 * (w @ Sigma @ w)
                - a2 * (r_hat @ w)
                - a3 * (epsilon @ w)
            ) * SCALE

        try:
            res = minimize(
                obj_mvf,
                init,
                method="SLSQP",
                bounds=bnds,
```

```
                constraints=cons
            )
            weights[name] = res.x
        except:
            weights[name] = init

    # ------------------------------------
    # REALIZED RETURN (DESIMAL)
    # ------------------------------------
    daily_act = (
        dc_full[dc_full["Date"] == today]
        .set_index("Stock")["Actual_Return"]
        .reindex(stocks_list)
        .fillna(0)
        .values
    )

    ret_today = {}
    for name in scenarios:
        ret_today[name] = daily_act @ weights[name]
        portfolio_return_mvf[name].append(ret_today[name])

    ret_bench = daily_act.mean()
    log_ret_bench.append(ret_bench)
    dates_backtest.append(today)

    # ------------------------------------
    # PRINT (SAMA PERSIS DENGAN MV)
    # ------------------------------------
    print("\n" + "="*60)
    print(f"{today.date()} | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)")

    print(
        "Return | "
        + " | ".join([f"{k}: {ret_today[k]:.2%}" for k in scenarios])
        + f" | Bench: {ret_bench:.2%}"
    )

# ========================================
# 5. FINAL OUTPUT
# ========================================
```

```
print("\n" + "="*60)
print("BACKTEST MVF (4 SKENARIO) SELESAI")
```

```
Loading data...
Done.


============================================================
2022-04-22 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.87% | MVF_S2_RiskUp: -1.89% | MVF_S3_RiskMore: -1.71% | MVF_S4_VeryRisk: -1.64% | Bench:
1.69%

============================================================
2022-04-29 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.00% | MVF_S2_RiskUp: 0.00% | MVF_S3_RiskMore: 0.00% | MVF_S4_VeryRisk: 0.00% | Bench: 2.43%

============================================================
2022-05-20 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 8.37% | MVF_S2_RiskUp: 7.20% | MVF_S3_RiskMore: 5.07% | MVF_S4_VeryRisk: 4.76% | Bench: 1.72%

============================================================
2022-05-27 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.94% | MVF_S2_RiskUp: -2.01% | MVF_S3_RiskMore: -1.51% | MVF_S4_VeryRisk: -0.09% | Bench:
1.31%

============================================================
2022-06-03 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 7.40% | MVF_S2_RiskUp: 7.40% | MVF_S3_RiskMore: 7.40% | MVF_S4_VeryRisk: 7.40% | Bench: 1.84%

============================================================
2022-06-10 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -7.37% | MVF_S2_RiskUp: -7.37% | MVF_S3_RiskMore: -7.37% | MVF_S4_VeryRisk: -7.37% | Bench: -
1.61%

============================================================
2022-06-17 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.56% | MVF_S2_RiskUp: 0.56% | MVF_S3_RiskMore: 0.56% | MVF_S4_VeryRisk: 0.56% | Bench: -2.8
7%

============================================================
2022-06-24 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.07% | MVF_S2_RiskUp: -1.21% | MVF_S3_RiskMore: -0.62% | MVF_S4_VeryRisk: -0.20% | Bench:
2.15%
```

```
============================================================
2022-07-01 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.00% | MVF_S2_RiskUp: 1.00% | MVF_S3_RiskMore: 0.06% | MVF_S4_VeryRisk: -1.64% | Bench: -4.6
9%

============================================================
2022-07-08 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.25% | MVF_S2_RiskUp: 6.25% | MVF_S3_RiskMore: 6.25% | MVF_S4_VeryRisk: 6.25% | Bench: -0.4
7%

============================================================
2022-07-15 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.11% | MVF_S2_RiskUp: -2.11% | MVF_S3_RiskMore: -2.11% | MVF_S4_VeryRisk: -2.11% | Bench: -
0.85%

============================================================
2022-07-22 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.81% | MVF_S2_RiskUp: 1.71% | MVF_S3_RiskMore: 2.00% | MVF_S4_VeryRisk: 2.22% | Bench: 4.12%

============================================================
2022-07-29 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.17% | MVF_S2_RiskUp: 6.17% | MVF_S3_RiskMore: 6.17% | MVF_S4_VeryRisk: 5.14% | Bench: 1.47%

============================================================
2022-08-05 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -3.69% | MVF_S2_RiskUp: -2.32% | MVF_S3_RiskMore: -2.15% | MVF_S4_VeryRisk: -0.71% | Bench:
0.72%

============================================================
2022-08-12 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.32% | MVF_S2_RiskUp: 1.32% | MVF_S3_RiskMore: 1.32% | MVF_S4_VeryRisk: 1.32% | Bench: 1.67%

============================================================
2022-08-19 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -6.14% | MVF_S2_RiskUp: -5.24% | MVF_S3_RiskMore: -3.99% | MVF_S4_VeryRisk: -3.06% | Bench: -
0.46%

============================================================
2022-08-26 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.56% | MVF_S2_RiskUp: 3.56% | MVF_S3_RiskMore: 2.64% | MVF_S4_VeryRisk: 2.32% | Bench: 1.03%
```

```
============================================================
2022-09-02 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 5.99% | MVF_S2_RiskUp: 5.99% | MVF_S3_RiskMore: 3.33% | MVF_S4_VeryRisk: 2.87% | Bench: 0.37%

============================================================
2022-09-09 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.40% | MVF_S2_RiskUp: -1.40% | MVF_S3_RiskMore: -1.40% | MVF_S4_VeryRisk: -1.40% | Bench:
1.22%

============================================================
2022-09-16 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.89% | MVF_S2_RiskUp: -0.89% | MVF_S3_RiskMore: 0.00% | MVF_S4_VeryRisk: -0.47% | Bench: 1.
94%

============================================================
2022-09-23 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.11% | MVF_S2_RiskUp: 3.11% | MVF_S3_RiskMore: 3.11% | MVF_S4_VeryRisk: 3.20% | Bench: -0.2
5%

============================================================
2022-09-30 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -6.46% | MVF_S2_RiskUp: -3.20% | MVF_S3_RiskMore: -2.80% | MVF_S4_VeryRisk: -2.49% | Bench: -
3.35%

============================================================
2022-10-07 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.73% | MVF_S2_RiskUp: 1.80% | MVF_S3_RiskMore: 0.63% | MVF_S4_VeryRisk: -0.25% | Bench: 0.8
9%

============================================================
2022-10-14 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -5.12% | MVF_S2_RiskUp: -1.72% | MVF_S3_RiskMore: 0.24% | MVF_S4_VeryRisk: 1.70% | Bench: -1.
45%

============================================================
2022-10-21 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 4.42% | MVF_S2_RiskUp: 3.98% | MVF_S3_RiskMore: 3.92% | MVF_S4_VeryRisk: 3.88% | Bench: 2.98%

============================================================
2022-10-28 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.00% | MVF_S2_RiskUp: 1.17% | MVF_S3_RiskMore: 1.88% | MVF_S4_VeryRisk: 3.04% | Bench: 1.35%
```

```
============================================================
2022-11-04 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.93% | MVF_S2_RiskUp: 3.93% | MVF_S3_RiskMore: 2.33% | MVF_S4_VeryRisk: -1.60% | Bench: -1.0
8%

============================================================
2022-11-11 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.66% | MVF_S2_RiskUp: -0.66% | MVF_S3_RiskMore: -0.66% | MVF_S4_VeryRisk: -0.78% | Bench: -
0.87%

============================================================
2022-11-18 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -7.76% | MVF_S2_RiskUp: -5.12% | MVF_S3_RiskMore: -3.15% | MVF_S4_VeryRisk: -1.67% | Bench: -
0.81%

============================================================
2022-11-25 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.40% | MVF_S2_RiskUp: -1.40% | MVF_S3_RiskMore: -1.40% | MVF_S4_VeryRisk: -1.40% | Bench:
1.12%

============================================================
2022-12-02 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.50% | MVF_S2_RiskUp: 6.52% | MVF_S3_RiskMore: 6.53% | MVF_S4_VeryRisk: 6.54% | Bench: 0.64%

============================================================
2022-12-09 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.46% | MVF_S2_RiskUp: -0.46% | MVF_S3_RiskMore: -1.40% | MVF_S4_VeryRisk: -1.76% | Bench: -
4.54%

============================================================
2022-12-16 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.40% | MVF_S2_RiskUp: 1.79% | MVF_S3_RiskMore: 1.39% | MVF_S4_VeryRisk: 1.08% | Bench: 0.77%

============================================================
2022-12-23 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.76% | MVF_S2_RiskUp: 0.00% | MVF_S3_RiskMore: -0.01% | MVF_S4_VeryRisk: -0.05% | Bench: -
0.32%

============================================================
2022-12-30 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
```

```
Return | MVF_S1_Equal: -2.15% | MVF_S2_RiskUp: -2.15% | MVF_S3_RiskMore: -1.05% | MVF_S4_VeryRisk: -0.26% | Bench: -
0.80%

============================================================
2023-01-06 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.04% | MVF_S2_RiskUp: -0.33% | MVF_S3_RiskMore: -0.38% | MVF_S4_VeryRisk: -0.41% | Bench: -
2.62%

============================================================
2023-01-13 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.24% | MVF_S2_RiskUp: 2.24% | MVF_S3_RiskMore: 2.26% | MVF_S4_VeryRisk: 2.40% | Bench: -0.7
1%

============================================================
2023-01-20 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.42% | MVF_S2_RiskUp: 0.42% | MVF_S3_RiskMore: 0.42% | MVF_S4_VeryRisk: 0.42% | Bench: 3.11%

============================================================
2023-01-27 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.15% | MVF_S2_RiskUp: 6.15% | MVF_S3_RiskMore: 6.15% | MVF_S4_VeryRisk: 3.52% | Bench: 2.33%

============================================================
2023-02-03 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -3.45% | MVF_S2_RiskUp: -2.03% | MVF_S3_RiskMore: -0.41% | MVF_S4_VeryRisk: 1.11% | Bench: 0.
43%

============================================================
2023-02-10 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.64% | MVF_S2_RiskUp: 2.92% | MVF_S3_RiskMore: 4.33% | MVF_S4_VeryRisk: 5.38% | Bench: 0.63%

============================================================
2023-02-17 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.46% | MVF_S2_RiskUp: -0.46% | MVF_S3_RiskMore: -0.46% | MVF_S4_VeryRisk: -0.46% | Bench: -
0.53%

============================================================
2023-02-24 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.35% | MVF_S2_RiskUp: -1.85% | MVF_S3_RiskMore: -1.67% | MVF_S4_VeryRisk: -1.54% | Bench: -
0.37%

============================================================
```

```
2023-03-03 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 12.73% | MVF_S2_RiskUp: 8.41% | MVF_S3_RiskMore: 7.15% | MVF_S4_VeryRisk: 5.36% | Bench: -0.2
2%

============================================================
2023-03-10 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.41% | MVF_S2_RiskUp: 0.52% | MVF_S3_RiskMore: 0.54% | MVF_S4_VeryRisk: 0.56% | Bench: -1.3
6%

============================================================
2023-03-17 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.26% | MVF_S2_RiskUp: -0.26% | MVF_S3_RiskMore: -2.06% | MVF_S4_VeryRisk: -3.64% | Bench: -
3.31%

============================================================
2023-03-24 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.56% | MVF_S2_RiskUp: 0.48% | MVF_S3_RiskMore: -0.22% | MVF_S4_VeryRisk: -0.75% | Bench: 1.4
1%

============================================================
2023-03-31 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.96% | MVF_S2_RiskUp: 2.96% | MVF_S3_RiskMore: 2.96% | MVF_S4_VeryRisk: 2.96% | Bench: 2.48%

============================================================
2023-04-07 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.35% | MVF_S2_RiskUp: -1.35% | MVF_S3_RiskMore: -1.35% | MVF_S4_VeryRisk: -2.89% | Bench: -
0.53%

============================================================
2023-04-14 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.09% | MVF_S2_RiskUp: -4.09% | MVF_S3_RiskMore: -4.09% | MVF_S4_VeryRisk: -3.68% | Bench:
0.49%

============================================================
2023-04-21 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.41% | MVF_S2_RiskUp: 0.41% | MVF_S3_RiskMore: 0.41% | MVF_S4_VeryRisk: 0.41% | Bench: -0.0
5%

============================================================
2023-04-28 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 15.18% | MVF_S2_RiskUp: 15.18% | MVF_S3_RiskMore: 15.18% | MVF_S4_VeryRisk: 14.61% | Bench:
```

```
2.30%

============================================================
2023-05-05 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.48% | MVF_S2_RiskUp: -1.48% | MVF_S3_RiskMore: -1.48% | MVF_S4_VeryRisk: -1.48% | Bench: -
2.54%

============================================================
2023-05-12 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.74% | MVF_S2_RiskUp: -1.92% | MVF_S3_RiskMore: -2.02% | MVF_S4_VeryRisk: -2.08% | Bench:
0.84%

============================================================
2023-05-19 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.79% | MVF_S2_RiskUp: 1.06% | MVF_S3_RiskMore: 1.05% | MVF_S4_VeryRisk: 1.04% | Bench: -1.7
1%

============================================================
2023-05-26 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.97% | MVF_S2_RiskUp: -0.97% | MVF_S3_RiskMore: -0.97% | MVF_S4_VeryRisk: -0.60% | Bench:
0.03%

============================================================
2023-06-02 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.48% | MVF_S2_RiskUp: -4.18% | MVF_S3_RiskMore: -4.09% | MVF_S4_VeryRisk: -3.22% | Bench: -
1.26%

============================================================
2023-06-09 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 11.43% | MVF_S2_RiskUp: 11.43% | MVF_S3_RiskMore: 11.43% | MVF_S4_VeryRisk: 7.91% | Bench: 4.
59%

============================================================
2023-06-16 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 5.21% | MVF_S2_RiskUp: 5.21% | MVF_S3_RiskMore: 5.21% | MVF_S4_VeryRisk: 5.21% | Bench: 0.71%

============================================================
2023-06-23 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.46% | MVF_S2_RiskUp: -3.28% | MVF_S3_RiskMore: -2.87% | MVF_S4_VeryRisk: -2.57% | Bench: -
0.51%
```

```
============================================================
2023-06-30 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.75% | MVF_S2_RiskUp: 0.75% | MVF_S3_RiskMore: 1.07% | MVF_S4_VeryRisk: 1.10% | Bench: 0.15%

============================================================
2023-07-07 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.39% | MVF_S2_RiskUp: 6.41% | MVF_S3_RiskMore: 6.28% | MVF_S4_VeryRisk: 5.64% | Bench: 2.00%

============================================================
2023-07-14 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 7.25% | MVF_S2_RiskUp: 7.25% | MVF_S3_RiskMore: 7.25% | MVF_S4_VeryRisk: 6.67% | Bench: 1.50%

============================================================
2023-07-21 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.24% | MVF_S2_RiskUp: 0.38% | MVF_S3_RiskMore: 0.23% | MVF_S4_VeryRisk: 0.03% | Bench: 0.60%

============================================================
2023-07-28 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 8.78% | MVF_S2_RiskUp: 8.78% | MVF_S3_RiskMore: 8.70% | MVF_S4_VeryRisk: 7.88% | Bench: -0.8
6%

============================================================
2023-08-04 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.09% | MVF_S2_RiskUp: -2.09% | MVF_S3_RiskMore: -2.23% | MVF_S4_VeryRisk: -3.15% | Bench: -
0.96%

============================================================
2023-08-11 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.94% | MVF_S2_RiskUp: 2.94% | MVF_S3_RiskMore: 2.60% | MVF_S4_VeryRisk: 1.93% | Bench: -0.2
9%

============================================================
2023-08-18 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.43% | MVF_S2_RiskUp: 1.43% | MVF_S3_RiskMore: 1.43% | MVF_S4_VeryRisk: 1.43% | Bench: 0.70%

============================================================
2023-08-25 | MEAN–VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 5.63% | MVF_S2_RiskUp: 5.63% | MVF_S3_RiskMore: 5.63% | MVF_S4_VeryRisk: 5.63% | Bench: -0.1
7%

============================================================
```

```
2023-09-01 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.33% | MVF_S2_RiskUp: -1.33% | MVF_S3_RiskMore: -1.33% | MVF_S4_VeryRisk: -1.30% | Bench:
1.01%

============================================================
2023-09-08 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.10% | MVF_S2_RiskUp: 3.10% | MVF_S3_RiskMore: 3.10% | MVF_S4_VeryRisk: 3.10% | Bench: 0.03%

============================================================
2023-09-15 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.39% | MVF_S2_RiskUp: 0.83% | MVF_S3_RiskMore: 0.60% | MVF_S4_VeryRisk: 0.43% | Bench: 1.10%

============================================================
2023-09-22 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.95% | MVF_S2_RiskUp: 1.94% | MVF_S3_RiskMore: 1.58% | MVF_S4_VeryRisk: 1.31% | Bench: 1.65%

============================================================
2023-09-29 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.22% | MVF_S2_RiskUp: -3.72% | MVF_S3_RiskMore: -4.31% | MVF_S4_VeryRisk: -4.38% | Bench: -
1.73%

============================================================
2023-10-06 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -5.96% | MVF_S2_RiskUp: -6.02% | MVF_S3_RiskMore: -6.15% | MVF_S4_VeryRisk: -6.25% | Bench: -
1.70%

============================================================
2023-10-13 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.34% | MVF_S2_RiskUp: 6.06% | MVF_S3_RiskMore: 5.31% | MVF_S4_VeryRisk: 5.00% | Bench: 1.16%

============================================================
2023-10-20 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 7.02% | MVF_S2_RiskUp: 7.02% | MVF_S3_RiskMore: 6.77% | MVF_S4_VeryRisk: 5.82% | Bench: -1.5
6%

============================================================
2023-10-27 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -11.25% | MVF_S2_RiskUp: -11.25% | MVF_S3_RiskMore: -11.06% | MVF_S4_VeryRisk: -8.91% | Benc
h: -1.63%

============================================================
```

```
2023-11-03 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.38% | MVF_S2_RiskUp: 6.38% | MVF_S3_RiskMore: 6.38% | MVF_S4_VeryRisk: 5.83% | Bench: -1.6
5%

============================================================
2023-11-10 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.13% | MVF_S2_RiskUp: 1.13% | MVF_S3_RiskMore: 1.13% | MVF_S4_VeryRisk: 1.07% | Bench: -1.3
0%

============================================================
2023-11-17 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.40% | MVF_S2_RiskUp: 0.40% | MVF_S3_RiskMore: 0.40% | MVF_S4_VeryRisk: 0.40% | Bench: 0.80%

============================================================
2023-11-24 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.71% | MVF_S2_RiskUp: -2.71% | MVF_S3_RiskMore: -1.62% | MVF_S4_VeryRisk: -0.26% | Bench:
0.10%

============================================================
2023-12-01 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.28% | MVF_S2_RiskUp: 3.28% | MVF_S3_RiskMore: 3.28% | MVF_S4_VeryRisk: 3.28% | Bench: 0.37%

============================================================
2023-12-08 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 61.51% | MVF_S2_RiskUp: 61.51% | MVF_S3_RiskMore: 61.51% | MVF_S4_VeryRisk: 61.51% | Bench: -
0.50%

============================================================
2023-12-15 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 4.89% | MVF_S2_RiskUp: 4.89% | MVF_S3_RiskMore: 4.89% | MVF_S4_VeryRisk: 4.89% | Bench: 0.26%

============================================================
2023-12-22 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 19.17% | MVF_S2_RiskUp: 19.17% | MVF_S3_RiskMore: 19.17% | MVF_S4_VeryRisk: 19.17% | Bench:
0.37%

============================================================
2023-12-29 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -10.64% | MVF_S2_RiskUp: -10.64% | MVF_S3_RiskMore: -10.64% | MVF_S4_VeryRisk: -10.64% | Benc
h: 1.12%
```

```
============================================================
2024-01-05 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 9.05% | MVF_S2_RiskUp: 9.05% | MVF_S3_RiskMore: 9.05% | MVF_S4_VeryRisk: 7.79% | Bench: 1.79%

============================================================
2024-01-12 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.42% | MVF_S2_RiskUp: -1.42% | MVF_S3_RiskMore: -1.42% | MVF_S4_VeryRisk: -1.42% | Bench: -
0.05%

============================================================
2024-01-19 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.98% | MVF_S2_RiskUp: -4.23% | MVF_S3_RiskMore: -4.38% | MVF_S4_VeryRisk: -3.89% | Bench: -
0.51%

============================================================
2024-01-26 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 24.14% | MVF_S2_RiskUp: 24.14% | MVF_S3_RiskMore: 24.14% | MVF_S4_VeryRisk: 24.14% | Bench: -
1.52%

============================================================
2024-02-02 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.39% | MVF_S2_RiskUp: 1.39% | MVF_S3_RiskMore: 1.39% | MVF_S4_VeryRisk: 1.39% | Bench: 0.21%

============================================================
2024-02-09 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -5.02% | MVF_S2_RiskUp: -5.02% | MVF_S3_RiskMore: -3.74% | MVF_S4_VeryRisk: -2.25% | Bench: -
0.11%

============================================================
2024-02-16 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -11.15% | MVF_S2_RiskUp: -11.15% | MVF_S3_RiskMore: -11.15% | MVF_S4_VeryRisk: -9.65% | Benc
h: 2.28%

============================================================
2024-02-23 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -9.82% | MVF_S2_RiskUp: -4.45% | MVF_S3_RiskMore: -3.20% | MVF_S4_VeryRisk: -1.49% | Bench: -
0.18%

============================================================
2024-03-01 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 4.01% | MVF_S2_RiskUp: 4.01% | MVF_S3_RiskMore: 4.01% | MVF_S4_VeryRisk: 4.01% | Bench: 0.61%
```

```
============================================================
2024-03-08 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.36% | MVF_S2_RiskUp: 2.08% | MVF_S3_RiskMore: 2.03% | MVF_S4_VeryRisk: 1.99% | Bench: 0.74%

============================================================
2024-03-15 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -7.62% | MVF_S2_RiskUp: -7.62% | MVF_S3_RiskMore: -7.62% | MVF_S4_VeryRisk: -7.62% | Bench: -
0.43%

============================================================
2024-03-22 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 10.19% | MVF_S2_RiskUp: 10.19% | MVF_S3_RiskMore: 10.19% | MVF_S4_VeryRisk: 9.69% | Bench: 2.
81%

============================================================
2024-03-29 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -7.10% | MVF_S2_RiskUp: -1.35% | MVF_S3_RiskMore: -0.32% | MVF_S4_VeryRisk: 0.45% | Bench: -
0.69%

============================================================
2024-04-05 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.53% | MVF_S2_RiskUp: -0.53% | MVF_S3_RiskMore: -0.67% | MVF_S4_VeryRisk: -1.17% | Bench: -
0.12%

============================================================
2024-04-26 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.93% | MVF_S2_RiskUp: 6.93% | MVF_S3_RiskMore: 6.93% | MVF_S4_VeryRisk: 6.93% | Bench: -1.6
3%

============================================================
2024-05-03 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 5.80% | MVF_S2_RiskUp: 5.80% | MVF_S3_RiskMore: 5.80% | MVF_S4_VeryRisk: 5.80% | Bench: 1.14%

============================================================
2024-05-10 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.33% | MVF_S2_RiskUp: 0.33% | MVF_S3_RiskMore: 0.33% | MVF_S4_VeryRisk: 0.13% | Bench: -0.7
4%

============================================================
2024-05-17 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
```

```
Return | MVF_S1_Equal: 3.99% | MVF_S2_RiskUp: 2.66% | MVF_S3_RiskMore: 1.85% | MVF_S4_VeryRisk: 1.25% | Bench: 2.42%

============================================================
2024-05-24 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.21% | MVF_S2_RiskUp: -0.21% | MVF_S3_RiskMore: -0.21% | MVF_S4_VeryRisk: -0.21% | Bench: -
0.67%

============================================================
2024-05-31 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.63% | MVF_S2_RiskUp: -0.63% | MVF_S3_RiskMore: -0.63% | MVF_S4_VeryRisk: -1.12% | Bench: -
3.06%

============================================================
2024-06-07 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.29% | MVF_S2_RiskUp: 6.29% | MVF_S3_RiskMore: 6.29% | MVF_S4_VeryRisk: 6.29% | Bench: 2.30%

============================================================
2024-06-14 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -5.84% | MVF_S2_RiskUp: -5.84% | MVF_S3_RiskMore: -5.06% | MVF_S4_VeryRisk: -4.18% | Bench: -
4.32%

============================================================
2024-06-21 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 5.33% | MVF_S2_RiskUp: 5.33% | MVF_S3_RiskMore: 5.33% | MVF_S4_VeryRisk: 5.33% | Bench: 1.99%

============================================================
2024-06-28 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 6.96% | MVF_S2_RiskUp: 5.28% | MVF_S3_RiskMore: 4.75% | MVF_S4_VeryRisk: 4.35% | Bench: 1.28%

============================================================
2024-07-05 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.27% | MVF_S2_RiskUp: -0.27% | MVF_S3_RiskMore: 0.06% | MVF_S4_VeryRisk: -0.37% | Bench: 2.
40%

============================================================
2024-07-12 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 3.26% | MVF_S2_RiskUp: 1.36% | MVF_S3_RiskMore: 0.41% | MVF_S4_VeryRisk: -0.30% | Bench: 2.9
4%

============================================================
2024-07-19 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
```

```
Return | MVF_S1_Equal: -4.33% | MVF_S2_RiskUp: -2.81% | MVF_S3_RiskMore: -2.12% | MVF_S4_VeryRisk: -0.98% | Bench:
0.19%

============================================================
2024-07-26 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.22% | MVF_S2_RiskUp: -0.99% | MVF_S3_RiskMore: -0.95% | MVF_S4_VeryRisk: -0.92% | Bench: -
0.85%

============================================================
2024-08-02 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.41% | MVF_S2_RiskUp: -2.41% | MVF_S3_RiskMore: -2.41% | MVF_S4_VeryRisk: -2.41% | Bench:
0.15%

============================================================
2024-08-09 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.95% | MVF_S2_RiskUp: -0.95% | MVF_S3_RiskMore: -0.95% | MVF_S4_VeryRisk: -0.95% | Bench: -
0.97%

============================================================
2024-08-16 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 5.79% | MVF_S2_RiskUp: 5.79% | MVF_S3_RiskMore: 5.79% | MVF_S4_VeryRisk: 5.55% | Bench: 3.33%

============================================================
2024-08-23 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.05% | MVF_S2_RiskUp: 1.87% | MVF_S3_RiskMore: 1.64% | MVF_S4_VeryRisk: 1.47% | Bench: 1.53%

============================================================
2024-08-30 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 1.55% | MVF_S2_RiskUp: 1.48% | MVF_S3_RiskMore: 1.47% | MVF_S4_VeryRisk: 1.22% | Bench: -0.4
7%

============================================================
2024-09-06 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.39% | MVF_S2_RiskUp: -0.39% | MVF_S3_RiskMore: -0.38% | MVF_S4_VeryRisk: -0.32% | Bench:
0.56%

============================================================
2024-09-13 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 2.77% | MVF_S2_RiskUp: 2.64% | MVF_S3_RiskMore: 2.62% | MVF_S4_VeryRisk: 2.60% | Bench: -0.3
4%
```

```
============================================================
2024-09-20 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.34% | MVF_S2_RiskUp: -1.10% | MVF_S3_RiskMore: -0.92% | MVF_S4_VeryRisk: -0.68% | Bench:
1.07%

============================================================
2024-09-27 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.58% | MVF_S2_RiskUp: -4.58% | MVF_S3_RiskMore: -4.65% | MVF_S4_VeryRisk: -3.96% | Bench:
1.82%

============================================================
2024-10-04 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.31% | MVF_S2_RiskUp: -2.54% | MVF_S3_RiskMore: -3.11% | MVF_S4_VeryRisk: -3.54% | Bench: -
1.61%

============================================================
2024-10-11 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -1.70% | MVF_S2_RiskUp: -1.34% | MVF_S3_RiskMore: -0.81% | MVF_S4_VeryRisk: -0.41% | Bench:
1.76%

============================================================
2024-10-18 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.33% | MVF_S2_RiskUp: -2.33% | MVF_S3_RiskMore: -2.33% | MVF_S4_VeryRisk: -2.33% | Bench:
1.53%

============================================================
2024-10-25 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.35% | MVF_S2_RiskUp: -4.35% | MVF_S3_RiskMore: -4.35% | MVF_S4_VeryRisk: -3.78% | Bench: -
2.22%

============================================================
2024-11-01 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.19% | MVF_S2_RiskUp: -0.09% | MVF_S3_RiskMore: -0.99% | MVF_S4_VeryRisk: -1.73% | Bench: -
2.88%

============================================================
2024-11-08 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -2.04% | MVF_S2_RiskUp: -2.04% | MVF_S3_RiskMore: -2.04% | MVF_S4_VeryRisk: -2.04% | Bench: -
3.00%

============================================================
```

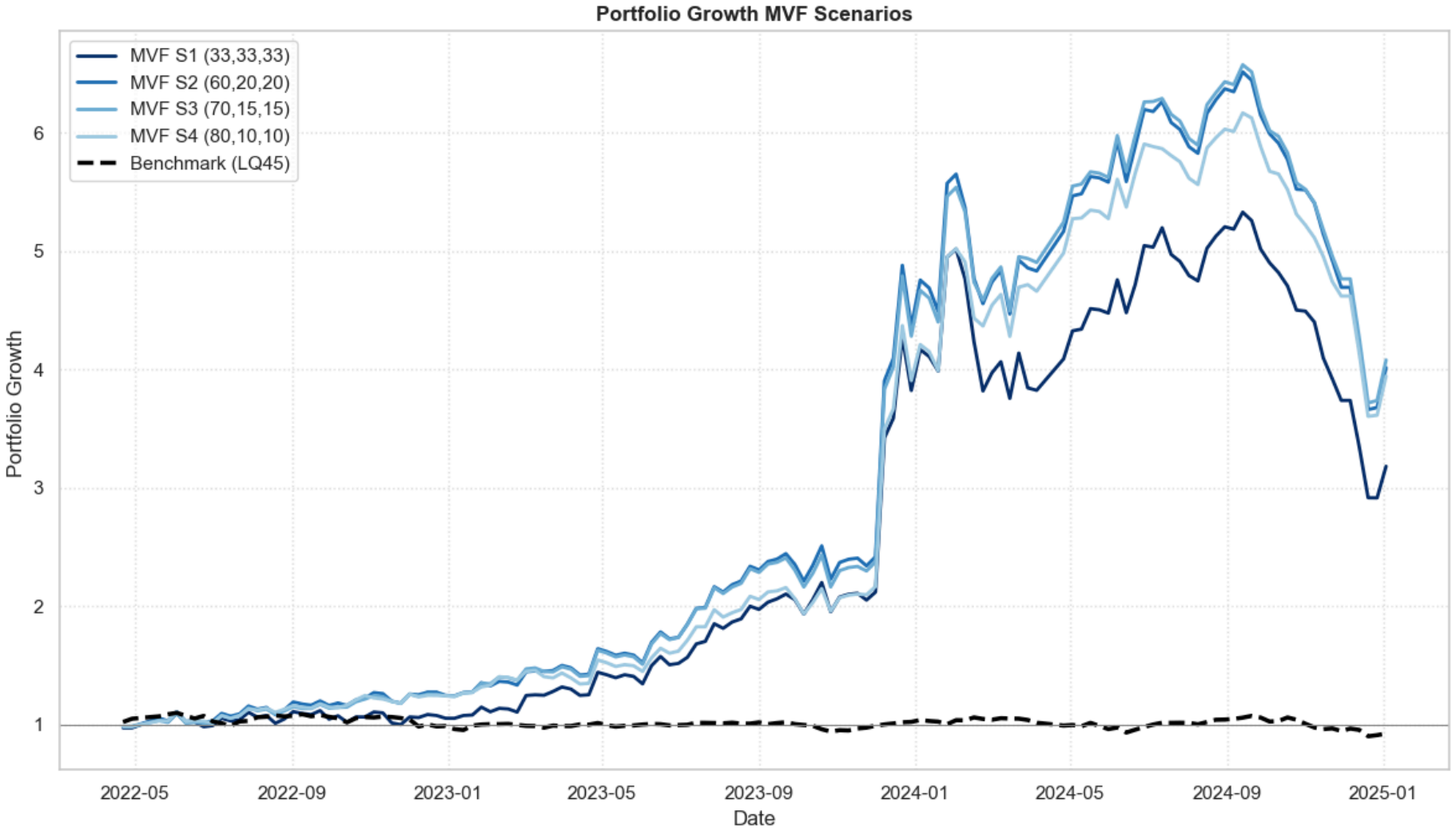
```
2024-11-15 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -6.93% | MVF_S2_RiskUp: -4.86% | MVF_S3_RiskMore: -4.24% | MVF_S4_VeryRisk: -3.12% | Bench: -
2.65%

============================================================
2024-11-22 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.27% | MVF_S2_RiskUp: -4.27% | MVF_S3_RiskMore: -4.27% | MVF_S4_VeryRisk: -4.26% | Bench: -
0.08%

============================================================
2024-11-29 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -4.65% | MVF_S2_RiskUp: -4.65% | MVF_S3_RiskMore: -3.89% | MVF_S4_VeryRisk: -2.58% | Bench: -
0.69%

============================================================
2024-12-06 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 0.00% | MVF_S2_RiskUp: 0.00% | MVF_S3_RiskMore: 0.00% | MVF_S4_VeryRisk: 0.00% | Bench: 2.41%

============================================================
2024-12-13 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -10.37% | MVF_S2_RiskUp: -10.37% | MVF_S3_RiskMore: -10.37% | MVF_S4_VeryRisk: -10.37% | Benc
h: 0.79%

============================================================
2024-12-20 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -12.97% | MVF_S2_RiskUp: -12.97% | MVF_S3_RiskMore: -12.97% | MVF_S4_VeryRisk: -12.97% | Benc
h: -5.99%

============================================================
2024-12-27 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: -0.04% | MVF_S2_RiskUp: 0.47% | MVF_S3_RiskMore: 0.56% | MVF_S4_VeryRisk: 0.26% | Bench: 1.9
4%

============================================================
2025-01-03 | MEAN-VARIANCE FORECAST ERROR (4 SKENARIO)
Return | MVF_S1_Equal: 9.15% | MVF_S2_RiskUp: 9.15% | MVF_S3_RiskMore: 9.15% | MVF_S4_VeryRisk: 9.15% | Bench: 0.61%

============================================================
BACKTEST MVF (4 SKENARIO) SELESAI
```

In [12]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# ========================================
# 1. PERSIAPAN DATA BENCHMARK LQ45
# ========================================
df_lq45 = pd.read_csv('JKLQ45_daily_2022_2025.csv')
df_lq45 = df_lq45.iloc[2:].copy()
df_lq45.columns = ['Date','Close','High','Low','Open','Volume']
df_lq45['Date'] = pd.to_datetime(df_lq45['Date'])
df_lq45['Close'] = pd.to_numeric(df_lq45['Close'])
df_lq45.set_index('Date', inplace=True)

# Return mingguan (Simple Return)
lq45_weekly_ret = df_lq45['Close'].resample('W-FRI').last().ffill().pct_change().fillna(0)

dates_bt = pd.to_datetime(dates_backtest).tz_localize(None)
lq45_aligned = lq45_weekly_ret.reindex(dates_bt, method='ffill').fillna(0).values

min_len_mvf = min(len(portfolio_return_mvf["MVF_S1_Equal"]), len(lq45_aligned))
dates_bt = dates_bt[-min_len_mvf:]
lq45_bench = lq45_aligned[-min_len_mvf:]

# ========================================
# 2. RETURN PER SKENARIO
# ========================================
scenarios_mvf = {
    "MVF S1 (33,33,33)": np.array(portfolio_return_mvf["MVF_S1_Equal"])[-min_len_mvf:],
    "MVF S2 (60,20,20)": np.array(portfolio_return_mvf["MVF_S2_RiskUp"])[-min_len_mvf:],
    "MVF S3 (70,15,15)": np.array(portfolio_return_mvf["MVF_S3_RiskMore"])[-min_len_mvf:],
    "MVF S4 (80,10,10)": np.array(portfolio_return_mvf["MVF_S4_VeryRisk"])[-min_len_mvf:],
    "Benchmark (LQ45)":  lq45_bench
}

# ========================================
# 3. FUNGSI METRIK
# ========================================
def get_performance_metrics(returns, benchmark, is_benchmark=False):
    ann_factor = 52
```

```
    excess_ret = returns - benchmark if not is_benchmark else returns

    sd_ann = np.std(excess_ret) * np.sqrt(ann_factor)

    if is_benchmark:
        ir_ann = np.nan
    else:
        er_ann = np.mean(excess_ret) * ann_factor
        ir_ann = er_ann / sd_ann if sd_ann != 0 else np.nan

    # Portfolio Growth (NAV)
    wealth_path = np.cumprod(1 + returns)
    portfolio_growth = wealth_path[-1]

    # Cumulative Return (%)
    cumulative_return = portfolio_growth - 1

    # Maximum Drawdown
    running_max = np.maximum.accumulate(wealth_path)
    md = ((wealth_path - running_max) / running_max).min()

    return [sd_ann, ir_ann, md, portfolio_growth, cumulative_return]

# ========================================
# 4. TABEL METRIK
# ========================================
metrics_mvf_list = []

for name, ret in scenarios_mvf.items():
    is_bench = "Benchmark" in name
    m = get_performance_metrics(ret, lq45_bench, is_benchmark=is_bench)

    metrics_mvf_list.append({
        "Model Strategy": name,
        "SD (Risk)": round(m[0], 4),
        "IR (Sharpe)": round(m[1], 4) if not np.isnan(m[1]) else "-",
        "Max Drawdown": f"{round(m[2]*100,2)}%",
        "Portfolio Growth (NAV)": round(m[3], 4),
        "Cumulative Return (%)": f"{round(m[4]*100,2)}%"
    })
```

```
df_mvf_final = pd.DataFrame(metrics_mvf_list)
print("\n=== PERFORMANCE METRICS (ANNUALIZED | MVF MODEL) ===")
print(df_mvf_final.to_string(index=False))

# ==========================================
# 5. VISUALISASI PORTFOLIO GROWTH
# ==========================================
plt.figure(figsize=(12,7))

blue_shades = ['#08306b', '#2171b5', '#6baed6', '#9ecae1']

idx = 0
for name, ret in scenarios_mvf.items():
    cum = np.cumprod(1 + ret)
    if "Benchmark" in name:
        plt.plot(dates_bt, cum, label=name, color='black', linestyle='--', linewidth=2.5)
    else:
        plt.plot(dates_bt, cum, label=name, color=blue_shades[idx % 4], linewidth=2)
        idx += 1

plt.title("Portfolio Growth MVF Scenarios", fontweight='bold')
plt.xlabel("Date")
plt.ylabel("Portfolio Growth ")
plt.legend()
plt.grid(True, linestyle=':', alpha=0.6)
plt.axhline(1, color='grey', linewidth=0.8)

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

```
=== PERFORMANCE METRICS (ANNUALIZED | MVF MODEL) ===
   Model Strategy  SD (Risk) IR (Sharpe) Max Drawdown  Portfolio Growth (NAV) Cumulative Return (%)
MVF S1 (33,33,33)     0.5376      1.0947       -45.3%                  3.1805               218.05%
MVF S2 (60,20,20)     0.5221       1.279       -43.8%                  4.0136               301.36%
MVF S3 (70,15,15)     0.5150      1.3014      -43.46%                  4.0784               307.84%
MVF S4 (80,10,10)     0.5016       1.297      -41.57%                  3.9426               294.26%
 Benchmark (LQ45)     0.1300           -      -18.01%                  0.9232                -7.68%
```



In [13]:
```
import pandas as pd
import numpy as np
from scipy.optimize import minimize

# ========================================
# 1. LOAD DATA
# ========================================
print("Loading data...")
```

```
dc = pd.read_csv("Hasil_Prediksi_Saham_15_2.csv")
dc["Date"] = pd.to_datetime(dc["Date"])
dc = dc.sort_values(["Stock", "Date"])

# hanya untuk penentuan tanggal backtest
dc_test = dc[dc["Set"] == "Test"].copy()
dc_test["Weekly_Error"] = dc_test["Actual_Return"] - dc_test["Prediksi_Return"]
dc_test["Rolling_Error_20"] = (
    dc_test.groupby("Stock")["Weekly_Error"]
    .transform(lambda x: x.rolling(15).mean().shift(1))
)
dc_clean = dc_test.dropna(subset=["Rolling_Error_20"]).copy()

dc_full = dc.copy()

stocks_list = sorted(dc_full["Stock"].unique())
n = len(stocks_list)

WINDOW_SIZE = 400
SCALE = 100000

print("Done.\n")

# ============================================
# 2. TANGGAL BACKTEST
# ============================================
pred_dates = np.sort(dc_clean["Date"].unique())

# ============================================
# 3. 4 SKENARIO MEAN–VARIANCE (ALPHA)
# ============================================
alphas = {
    "MV_S1_Equal":   0.50,  # risiko = return
    "MV_S2_RiskUp":  0.65,
    "MV_S3_RiskMore":0.80,
    "MV_S4_MinVar":  0.95   # hampir pure risk minimization
}

portfolio_return_mv = {k: [] for k in alphas}
log_ret_bench = []
```

```
dates_backtest = []

weights_log = []

# ==========================================
# 4. BACKTEST
# ==========================================
for today in pred_dates:

    hist_data = (
        dc_full[dc_full["Date"] < today]
        .sort_values("Date")
        .tail(WINDOW_SIZE * n)
    )

    if hist_data["Date"].nunique() < WINDOW_SIZE:
        continue

    pivot_train = (
        hist_data.pivot(index="Date", columns="Stock", values="Actual_Return")
        .reindex(columns=stocks_list)
        .fillna(0)
    )

    Sigma = pivot_train.cov().values
    r = pivot_train.mean().values

    cons = ({'type': 'eq', 'fun': lambda w: np.sum(w) - 1})
    bnds = tuple((0.0, 1.0) for _ in range(n))
    init = np.ones(n) / n

    def obj_mv(w, Sigma, r, alpha):
        return (
            0.5 * alpha * (w @ Sigma @ w)
            - (1 - alpha) * (r @ w)
        ) * SCALE


    weights = {}

    for name, alpha in alphas.items():
```

```
        try:
            res = minimize(
                obj_mv,
                init,
                args=(Sigma, r, alpha),
                method="SLSQP",
                bounds=bnds,
                constraints=cons
            )
            weights[name] = res.x
        except:
            weights[name] = init

    # ------------------------------------
    # REALIZED RETURN
    # ------------------------------------
    daily_act = (
        dc_full[dc_full["Date"] == today]
        .set_index("Stock")["Actual_Return"]
        .reindex(stocks_list)
        .fillna(0)
        .values
    )

    ret_today = {}
    for name in alphas:
        ret_today[name] = daily_act @ weights[name]
        portfolio_return_mv[name].append(ret_today[name])

    ret_bench = daily_act.mean()
    log_ret_bench.append(ret_bench)
    dates_backtest.append(today)

    # ------------------------------------
    # SAVE WEIGHTS
    # ------------------------------------
    df_weights = pd.DataFrame({
        "Date": today,
        "Stock": stocks_list,
        **{f"W_{k}": v for k, v in weights.items()}
    })
```

```
    weights_log.append(df_weights)

    # ------------------------------------
    # PRINT RINGKAS
    # ------------------------------------
    print("\n" + "="*60)
    print(f"{today.date()} | MEAN-VARIANCE (4 SKENARIO)")

    print(
        "Return | "
        + " | ".join([f"{k}: {ret_today[k]:.2%}" for k in alphas])
        + f" | Bench: {ret_bench:.2%}"
    )

# ========================================
# 5. FINAL OUTPUT
# ========================================
weights_log = pd.concat(weights_log, ignore_index=True)

print("\n" + "="*60)
print("BACKTEST MEAN-VARIANCE (4 SKENARIO) SELESAI")
```

```
Loading data...
Done.


============================================================
2022-04-22 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.73% | MV_S2_RiskUp: -0.85% | MV_S3_RiskMore: -0.41% | MV_S4_MinVar: 0.55% | Bench: 1.69%

============================================================
2022-04-29 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.61% | MV_S2_RiskUp: -1.54% | MV_S3_RiskMore: -0.79% | MV_S4_MinVar: 1.05% | Bench: 2.43%

============================================================
2022-05-20 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.06% | MV_S2_RiskUp: 1.29% | MV_S3_RiskMore: 3.35% | MV_S4_MinVar: 2.64% | Bench: 1.72%

============================================================
2022-05-27 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.72% | MV_S2_RiskUp: 1.57% | MV_S3_RiskMore: 0.31% | MV_S4_MinVar: 1.46% | Bench: 1.31%

============================================================
2022-06-03 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.25% | MV_S2_RiskUp: 1.35% | MV_S3_RiskMore: 2.31% | MV_S4_MinVar: 2.07% | Bench: 1.84%

============================================================
2022-06-10 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.43% | MV_S2_RiskUp: -2.39% | MV_S3_RiskMore: -1.97% | MV_S4_MinVar: -1.55% | Bench: -1.61%

============================================================
2022-06-17 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.23% | MV_S2_RiskUp: -5.49% | MV_S3_RiskMore: -4.32% | MV_S4_MinVar: -0.90% | Bench: -2.87%

============================================================
2022-06-24 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.62% | MV_S2_RiskUp: -0.31% | MV_S3_RiskMore: 1.70% | MV_S4_MinVar: 3.43% | Bench: 2.15%

============================================================
2022-07-01 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -8.38% | MV_S2_RiskUp: -7.73% | MV_S3_RiskMore: -4.98% | MV_S4_MinVar: -2.70% | Bench: -4.69%

============================================================
```

```
2022-07-08 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.64% | MV_S2_RiskUp: 0.19% | MV_S3_RiskMore: -0.98% | MV_S4_MinVar: -1.11% | Bench: -0.47%

============================================================
2022-07-15 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 7.25% | MV_S2_RiskUp: 6.36% | MV_S3_RiskMore: 3.43% | MV_S4_MinVar: 1.14% | Bench: -0.85%

============================================================
2022-07-22 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.53% | MV_S2_RiskUp: 2.77% | MV_S3_RiskMore: 2.30% | MV_S4_MinVar: 2.07% | Bench: 4.12%

============================================================
2022-07-29 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.19% | MV_S2_RiskUp: 0.40% | MV_S3_RiskMore: -0.69% | MV_S4_MinVar: -1.60% | Bench: 1.47%

============================================================
2022-08-05 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.36% | MV_S2_RiskUp: -1.55% | MV_S3_RiskMore: -3.79% | MV_S4_MinVar: -1.29% | Bench: 0.72%

============================================================
2022-08-12 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.16% | MV_S2_RiskUp: 0.06% | MV_S3_RiskMore: 0.52% | MV_S4_MinVar: -0.40% | Bench: 1.67%

============================================================
2022-08-19 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.63% | MV_S2_RiskUp: -0.64% | MV_S3_RiskMore: 1.43% | MV_S4_MinVar: 1.84% | Bench: -0.46%

============================================================
2022-08-26 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 3.61% | MV_S2_RiskUp: 3.32% | MV_S3_RiskMore: 1.02% | MV_S4_MinVar: -0.45% | Bench: 1.03%

============================================================
2022-09-02 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.55% | MV_S2_RiskUp: 1.75% | MV_S3_RiskMore: 2.14% | MV_S4_MinVar: 1.36% | Bench: 0.37%

============================================================
2022-09-09 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 9.13% | MV_S2_RiskUp: 8.89% | MV_S3_RiskMore: 6.62% | MV_S4_MinVar: 2.47% | Bench: 1.22%

============================================================
2022-09-16 | MEAN–VARIANCE (4 SKENARIO)
```

```
Return | MV_S1_Equal: -4.72% | MV_S2_RiskUp: -4.56% | MV_S3_RiskMore: -1.94% | MV_S4_MinVar: 0.03% | Bench: 1.94%

============================================================
2022-09-23 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 4.68% | MV_S2_RiskUp: 4.67% | MV_S3_RiskMore: 4.49% | MV_S4_MinVar: 2.49% | Bench: -0.25%

============================================================
2022-09-30 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -7.05% | MV_S2_RiskUp: -6.40% | MV_S3_RiskMore: -4.49% | MV_S4_MinVar: -1.65% | Bench: -3.35%

============================================================
2022-10-07 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 5.25% | MV_S2_RiskUp: 4.30% | MV_S3_RiskMore: 2.54% | MV_S4_MinVar: -0.48% | Bench: 0.89%

============================================================
2022-10-14 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.87% | MV_S2_RiskUp: -2.13% | MV_S3_RiskMore: -0.90% | MV_S4_MinVar: -0.53% | Bench: -1.45%

============================================================
2022-10-21 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.72% | MV_S2_RiskUp: 1.93% | MV_S3_RiskMore: 2.38% | MV_S4_MinVar: 2.97% | Bench: 2.98%

============================================================
2022-10-28 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.75% | MV_S2_RiskUp: -1.06% | MV_S3_RiskMore: 0.53% | MV_S4_MinVar: 2.39% | Bench: 1.35%

============================================================
2022-11-04 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.43% | MV_S2_RiskUp: -3.46% | MV_S3_RiskMore: -3.25% | MV_S4_MinVar: -1.89% | Bench: -1.08%

============================================================
2022-11-11 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.78% | MV_S2_RiskUp: -2.26% | MV_S3_RiskMore: -1.69% | MV_S4_MinVar: -0.24% | Bench: -0.87%

============================================================
2022-11-18 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.68% | MV_S2_RiskUp: 2.64% | MV_S3_RiskMore: 3.23% | MV_S4_MinVar: 1.72% | Bench: -0.81%

============================================================
2022-11-25 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.74% | MV_S2_RiskUp: -1.03% | MV_S3_RiskMore: -0.51% | MV_S4_MinVar: 0.81% | Bench: 1.12%
```

```
============================================================
2022-12-02 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.62% | MV_S2_RiskUp: 2.39% | MV_S3_RiskMore: 2.92% | MV_S4_MinVar: 2.01% | Bench: 0.64%

============================================================
2022-12-09 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.10% | MV_S2_RiskUp: -3.71% | MV_S3_RiskMore: -5.35% | MV_S4_MinVar: -5.20% | Bench: -4.54%

============================================================
2022-12-16 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 4.77% | MV_S2_RiskUp: 4.69% | MV_S3_RiskMore: 3.90% | MV_S4_MinVar: 1.86% | Bench: 0.77%

============================================================
2022-12-23 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.96% | MV_S2_RiskUp: -0.61% | MV_S3_RiskMore: -0.18% | MV_S4_MinVar: -0.38% | Bench: -0.32%

============================================================
2022-12-30 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.43% | MV_S2_RiskUp: 1.69% | MV_S3_RiskMore: 1.58% | MV_S4_MinVar: 0.83% | Bench: -0.80%

============================================================
2023-01-06 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -9.39% | MV_S2_RiskUp: -8.57% | MV_S3_RiskMore: -5.47% | MV_S4_MinVar: -2.41% | Bench: -2.62%

============================================================
2023-01-13 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.60% | MV_S2_RiskUp: 1.10% | MV_S3_RiskMore: 0.25% | MV_S4_MinVar: -0.97% | Bench: -0.71%

============================================================
2023-01-20 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.80% | MV_S2_RiskUp: 0.60% | MV_S3_RiskMore: 0.63% | MV_S4_MinVar: 1.26% | Bench: 3.11%

============================================================
2023-01-27 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.95% | MV_S2_RiskUp: -1.31% | MV_S3_RiskMore: -0.06% | MV_S4_MinVar: 1.27% | Bench: 2.33%

============================================================
2023-02-03 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.70% | MV_S2_RiskUp: -1.90% | MV_S3_RiskMore: 0.05% | MV_S4_MinVar: 1.68% | Bench: 0.43%
```

```
============================================================
2023-02-10 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.13% | MV_S2_RiskUp: -0.02% | MV_S3_RiskMore: 0.03% | MV_S4_MinVar: 0.44% | Bench: 0.63%

============================================================
2023-02-17 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.42% | MV_S2_RiskUp: -0.38% | MV_S3_RiskMore: 1.10% | MV_S4_MinVar: 1.29% | Bench: -0.53%

============================================================
2023-02-24 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.37% | MV_S2_RiskUp: -0.65% | MV_S3_RiskMore: -0.91% | MV_S4_MinVar: -0.63% | Bench: -0.37%

============================================================
2023-03-03 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.60% | MV_S2_RiskUp: -1.51% | MV_S3_RiskMore: -1.05% | MV_S4_MinVar: -0.99% | Bench: -0.22%

============================================================
2023-03-10 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.11% | MV_S2_RiskUp: 1.69% | MV_S3_RiskMore: 1.36% | MV_S4_MinVar: 0.86% | Bench: -1.36%

============================================================
2023-03-17 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.77% | MV_S2_RiskUp: -3.07% | MV_S3_RiskMore: -3.05% | MV_S4_MinVar: -2.32% | Bench: -3.31%

============================================================
2023-03-24 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.91% | MV_S2_RiskUp: -0.40% | MV_S3_RiskMore: 0.35% | MV_S4_MinVar: 2.16% | Bench: 1.41%

============================================================
2023-03-31 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 7.85% | MV_S2_RiskUp: 5.85% | MV_S3_RiskMore: 4.03% | MV_S4_MinVar: 1.90% | Bench: 2.48%

============================================================
2023-04-07 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.40% | MV_S2_RiskUp: -0.54% | MV_S3_RiskMore: -1.83% | MV_S4_MinVar: -2.07% | Bench: -0.53%

============================================================
2023-04-14 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.05% | MV_S2_RiskUp: -0.69% | MV_S3_RiskMore: -0.56% | MV_S4_MinVar: 0.78% | Bench: 0.49%

============================================================
```

```
2023-04-21 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 3.39% | MV_S2_RiskUp: 2.61% | MV_S3_RiskMore: 1.22% | MV_S4_MinVar: 0.33% | Bench: -0.05%

============================================================
2023-04-28 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.64% | MV_S2_RiskUp: 1.22% | MV_S3_RiskMore: 2.05% | MV_S4_MinVar: 1.98% | Bench: 2.30%

============================================================
2023-05-05 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.88% | MV_S2_RiskUp: -5.15% | MV_S3_RiskMore: -3.50% | MV_S4_MinVar: -1.73% | Bench: -2.54%

============================================================
2023-05-12 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.60% | MV_S2_RiskUp: -2.19% | MV_S3_RiskMore: -2.21% | MV_S4_MinVar: -1.78% | Bench: 0.84%

============================================================
2023-05-19 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.22% | MV_S2_RiskUp: -4.81% | MV_S3_RiskMore: -3.70% | MV_S4_MinVar: -1.00% | Bench: -1.71%

============================================================
2023-05-26 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -4.08% | MV_S2_RiskUp: -4.13% | MV_S3_RiskMore: -3.51% | MV_S4_MinVar: -1.17% | Bench: 0.03%

============================================================
2023-06-02 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.29% | MV_S2_RiskUp: -0.16% | MV_S3_RiskMore: -1.28% | MV_S4_MinVar: -0.97% | Bench: -1.26%

============================================================
2023-06-09 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.54% | MV_S2_RiskUp: 0.73% | MV_S3_RiskMore: 1.12% | MV_S4_MinVar: 1.20% | Bench: 4.59%

============================================================
2023-06-16 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.24% | MV_S2_RiskUp: 1.55% | MV_S3_RiskMore: 0.14% | MV_S4_MinVar: -0.77% | Bench: 0.71%

============================================================
2023-06-23 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.98% | MV_S2_RiskUp: -0.61% | MV_S3_RiskMore: -0.44% | MV_S4_MinVar: -0.08% | Bench: -0.51%

============================================================
2023-06-30 | MEAN–VARIANCE (4 SKENARIO)
```

```
Return | MV_S1_Equal: 0.29% | MV_S2_RiskUp: 0.16% | MV_S3_RiskMore: -0.01% | MV_S4_MinVar: 0.41% | Bench: 0.15%

============================================================
2023-07-07 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.66% | MV_S2_RiskUp: 2.13% | MV_S3_RiskMore: 1.76% | MV_S4_MinVar: 0.62% | Bench: 2.00%

============================================================
2023-07-14 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.67% | MV_S2_RiskUp: -0.41% | MV_S3_RiskMore: -0.14% | MV_S4_MinVar: 0.93% | Bench: 1.50%

============================================================
2023-07-21 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.45% | MV_S2_RiskUp: 1.00% | MV_S3_RiskMore: 0.88% | MV_S4_MinVar: -0.03% | Bench: 0.60%

============================================================
2023-07-28 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 3.26% | MV_S2_RiskUp: 2.42% | MV_S3_RiskMore: 0.25% | MV_S4_MinVar: -2.07% | Bench: -0.86%

============================================================
2023-08-04 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -4.19% | MV_S2_RiskUp: -2.91% | MV_S3_RiskMore: -1.07% | MV_S4_MinVar: 0.42% | Bench: -0.96%

============================================================
2023-08-11 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.64% | MV_S2_RiskUp: 1.57% | MV_S3_RiskMore: 1.25% | MV_S4_MinVar: 1.53% | Bench: -0.29%

============================================================
2023-08-18 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.28% | MV_S2_RiskUp: 0.86% | MV_S3_RiskMore: 0.95% | MV_S4_MinVar: 0.61% | Bench: 0.70%

============================================================
2023-08-25 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.04% | MV_S2_RiskUp: 0.13% | MV_S3_RiskMore: 0.29% | MV_S4_MinVar: -0.07% | Bench: -0.17%

============================================================
2023-09-01 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.45% | MV_S2_RiskUp: 0.63% | MV_S3_RiskMore: 0.68% | MV_S4_MinVar: -1.07% | Bench: 1.01%

============================================================
2023-09-08 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.50% | MV_S2_RiskUp: -1.77% | MV_S3_RiskMore: -1.53% | MV_S4_MinVar: -1.32% | Bench: 0.03%
```

```
============================================================
2023-09-15 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 11.26% | MV_S2_RiskUp: 14.98% | MV_S3_RiskMore: 12.49% | MV_S4_MinVar: 3.90% | Bench: 1.10%

============================================================
2023-09-22 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 5.04% | MV_S2_RiskUp: 3.65% | MV_S3_RiskMore: 1.83% | MV_S4_MinVar: 1.24% | Bench: 1.65%

============================================================
2023-09-29 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.09% | MV_S2_RiskUp: -2.25% | MV_S3_RiskMore: -1.37% | MV_S4_MinVar: -1.60% | Bench: -1.73%

============================================================
2023-10-06 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -4.74% | MV_S2_RiskUp: -2.14% | MV_S3_RiskMore: -2.73% | MV_S4_MinVar: -0.12% | Bench: -1.70%

============================================================
2023-10-13 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.74% | MV_S2_RiskUp: 0.99% | MV_S3_RiskMore: 1.43% | MV_S4_MinVar: -0.39% | Bench: 1.16%

============================================================
2023-10-20 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.71% | MV_S2_RiskUp: -0.94% | MV_S3_RiskMore: -0.37% | MV_S4_MinVar: -1.07% | Bench: -1.56%

============================================================
2023-10-27 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.16% | MV_S2_RiskUp: -0.71% | MV_S3_RiskMore: 0.27% | MV_S4_MinVar: 0.59% | Bench: -1.63%

============================================================
2023-11-03 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.16% | MV_S2_RiskUp: -2.80% | MV_S3_RiskMore: -0.86% | MV_S4_MinVar: -0.90% | Bench: -1.65%

============================================================
2023-11-10 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.13% | MV_S2_RiskUp: -2.54% | MV_S3_RiskMore: -1.71% | MV_S4_MinVar: -0.77% | Bench: -1.30%

============================================================
2023-11-17 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.43% | MV_S2_RiskUp: -1.40% | MV_S3_RiskMore: -0.87% | MV_S4_MinVar: 0.55% | Bench: 0.80%
```

```
============================================================
2023-11-24 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.50% | MV_S2_RiskUp: -0.93% | MV_S3_RiskMore: -1.11% | MV_S4_MinVar: -0.94% | Bench: 0.10%

============================================================
2023-12-01 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.58% | MV_S2_RiskUp: -0.01% | MV_S3_RiskMore: 0.46% | MV_S4_MinVar: 0.43% | Bench: 0.37%

============================================================
2023-12-08 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 35.30% | MV_S2_RiskUp: 36.00% | MV_S3_RiskMore: 25.93% | MV_S4_MinVar: 10.46% | Bench: -0.50%

============================================================
2023-12-15 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.74% | MV_S2_RiskUp: 3.24% | MV_S3_RiskMore: 2.19% | MV_S4_MinVar: 1.51% | Bench: 0.26%

============================================================
2023-12-22 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 15.20% | MV_S2_RiskUp: 13.28% | MV_S3_RiskMore: 9.76% | MV_S4_MinVar: 3.90% | Bench: 0.37%

============================================================
2023-12-29 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -9.05% | MV_S2_RiskUp: -7.01% | MV_S3_RiskMore: -3.98% | MV_S4_MinVar: -0.96% | Bench: 1.12%

============================================================
2024-01-05 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 9.19% | MV_S2_RiskUp: 7.95% | MV_S3_RiskMore: 4.22% | MV_S4_MinVar: 1.96% | Bench: 1.79%

============================================================
2024-01-12 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -32.20% | MV_S2_RiskUp: -26.32% | MV_S3_RiskMore: -17.52% | MV_S4_MinVar: -5.71% | Bench: -0.0
5%

============================================================
2024-01-19 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 15.51% | MV_S2_RiskUp: 12.17% | MV_S3_RiskMore: 7.45% | MV_S4_MinVar: 1.72% | Bench: -0.51%

============================================================
2024-01-26 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 17.41% | MV_S2_RiskUp: 13.41% | MV_S3_RiskMore: 7.88% | MV_S4_MinVar: 1.45% | Bench: -1.52%
```

```
============================================================
2024-02-02 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.68% | MV_S2_RiskUp: -0.03% | MV_S3_RiskMore: 0.67% | MV_S4_MinVar: 1.58% | Bench: 0.21%

============================================================
2024-02-09 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.94% | MV_S2_RiskUp: -2.73% | MV_S3_RiskMore: -1.69% | MV_S4_MinVar: -0.83% | Bench: -0.11%

============================================================
2024-02-16 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -8.48% | MV_S2_RiskUp: -5.60% | MV_S3_RiskMore: -1.68% | MV_S4_MinVar: 2.13% | Bench: 2.28%

============================================================
2024-02-23 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.21% | MV_S2_RiskUp: -2.31% | MV_S3_RiskMore: -2.24% | MV_S4_MinVar: -0.06% | Bench: -0.18%

============================================================
2024-03-01 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 5.08% | MV_S2_RiskUp: 4.32% | MV_S3_RiskMore: 2.83% | MV_S4_MinVar: 0.59% | Bench: 0.61%

============================================================
2024-03-08 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 14.96% | MV_S2_RiskUp: 12.20% | MV_S3_RiskMore: 8.73% | MV_S4_MinVar: 4.22% | Bench: 0.74%

============================================================
2024-03-15 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.06% | MV_S2_RiskUp: -3.85% | MV_S3_RiskMore: -2.04% | MV_S4_MinVar: -0.72% | Bench: -0.43%

============================================================
2024-03-22 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 7.94% | MV_S2_RiskUp: 6.37% | MV_S3_RiskMore: 4.23% | MV_S4_MinVar: 2.02% | Bench: 2.81%

============================================================
2024-03-29 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.36% | MV_S2_RiskUp: 1.31% | MV_S3_RiskMore: 0.74% | MV_S4_MinVar: 0.96% | Bench: -0.69%

============================================================
2024-04-05 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 20.09% | MV_S2_RiskUp: 15.24% | MV_S3_RiskMore: 8.54% | MV_S4_MinVar: 2.31% | Bench: -0.12%

============================================================
```

```
2024-04-26 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 4.97% | MV_S2_RiskUp: 2.52% | MV_S3_RiskMore: 1.11% | MV_S4_MinVar: 1.87% | Bench: -1.63%

============================================================
2024-05-03 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 4.39% | MV_S2_RiskUp: 3.04% | MV_S3_RiskMore: 2.62% | MV_S4_MinVar: 2.41% | Bench: 1.14%

============================================================
2024-05-10 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.55% | MV_S2_RiskUp: 1.84% | MV_S3_RiskMore: 0.44% | MV_S4_MinVar: -1.48% | Bench: -0.74%

============================================================
2024-05-17 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 12.55% | MV_S2_RiskUp: 8.68% | MV_S3_RiskMore: 4.95% | MV_S4_MinVar: 2.03% | Bench: 2.42%

============================================================
2024-05-24 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.36% | MV_S2_RiskUp: -0.18% | MV_S3_RiskMore: -0.21% | MV_S4_MinVar: -0.81% | Bench: -0.67%

============================================================
2024-05-31 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.57% | MV_S2_RiskUp: 1.01% | MV_S3_RiskMore: -0.22% | MV_S4_MinVar: -2.25% | Bench: -3.06%

============================================================
2024-06-07 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -6.95% | MV_S2_RiskUp: -5.09% | MV_S3_RiskMore: -1.77% | MV_S4_MinVar: 2.29% | Bench: 2.30%

============================================================
2024-06-14 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.50% | MV_S2_RiskUp: -0.96% | MV_S3_RiskMore: -1.40% | MV_S4_MinVar: -1.44% | Bench: -4.32%

============================================================
2024-06-21 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.95% | MV_S2_RiskUp: 0.63% | MV_S3_RiskMore: 0.67% | MV_S4_MinVar: 0.63% | Bench: 1.99%

============================================================
2024-06-28 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 5.89% | MV_S2_RiskUp: 4.39% | MV_S3_RiskMore: 3.72% | MV_S4_MinVar: 3.43% | Bench: 1.28%

============================================================
2024-07-05 | MEAN–VARIANCE (4 SKENARIO)
```

```
Return | MV_S1_Equal: 0.93% | MV_S2_RiskUp: 1.78% | MV_S3_RiskMore: 1.31% | MV_S4_MinVar: 0.40% | Bench: 2.40%

============================================================
2024-07-12 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.82% | MV_S2_RiskUp: 1.52% | MV_S3_RiskMore: 1.69% | MV_S4_MinVar: 2.09% | Bench: 2.94%

============================================================
2024-07-19 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -0.57% | MV_S2_RiskUp: -0.53% | MV_S3_RiskMore: -0.27% | MV_S4_MinVar: 0.47% | Bench: 0.19%

============================================================
2024-07-26 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.19% | MV_S2_RiskUp: 0.64% | MV_S3_RiskMore: 0.10% | MV_S4_MinVar: 0.02% | Bench: -0.85%

============================================================
2024-08-02 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.99% | MV_S2_RiskUp: 1.00% | MV_S3_RiskMore: 0.18% | MV_S4_MinVar: 0.08% | Bench: 0.15%

============================================================
2024-08-09 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.14% | MV_S2_RiskUp: 2.27% | MV_S3_RiskMore: 1.25% | MV_S4_MinVar: -0.13% | Bench: -0.97%

============================================================
2024-08-16 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.42% | MV_S2_RiskUp: -0.56% | MV_S3_RiskMore: 0.25% | MV_S4_MinVar: 0.96% | Bench: 3.33%

============================================================
2024-08-23 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.48% | MV_S2_RiskUp: -3.97% | MV_S3_RiskMore: -2.39% | MV_S4_MinVar: -0.56% | Bench: 1.53%

============================================================
2024-08-30 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 3.12% | MV_S2_RiskUp: 2.50% | MV_S3_RiskMore: 1.52% | MV_S4_MinVar: 0.11% | Bench: -0.47%

============================================================
2024-09-06 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -3.91% | MV_S2_RiskUp: -2.69% | MV_S3_RiskMore: -1.55% | MV_S4_MinVar: -0.35% | Bench: 0.56%

============================================================
2024-09-13 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.71% | MV_S2_RiskUp: 0.48% | MV_S3_RiskMore: 1.56% | MV_S4_MinVar: 1.41% | Bench: -0.34%
```

```
============================================================
2024-09-20 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -4.91% | MV_S2_RiskUp: -3.93% | MV_S3_RiskMore: -1.90% | MV_S4_MinVar: 0.70% | Bench: 1.07%

============================================================
2024-09-27 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 7.71% | MV_S2_RiskUp: 6.90% | MV_S3_RiskMore: 3.52% | MV_S4_MinVar: 1.74% | Bench: 1.82%

============================================================
2024-10-04 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 0.83% | MV_S2_RiskUp: 0.31% | MV_S3_RiskMore: -0.89% | MV_S4_MinVar: -2.27% | Bench: -1.61%

============================================================
2024-10-11 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -1.68% | MV_S2_RiskUp: -1.17% | MV_S3_RiskMore: -0.67% | MV_S4_MinVar: -0.06% | Bench: 1.76%

============================================================
2024-10-18 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.23% | MV_S2_RiskUp: 2.95% | MV_S3_RiskMore: 3.31% | MV_S4_MinVar: 3.61% | Bench: 1.53%

============================================================
2024-10-25 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.23% | MV_S2_RiskUp: 1.61% | MV_S3_RiskMore: 0.97% | MV_S4_MinVar: -0.55% | Bench: -2.22%

============================================================
2024-11-01 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.79% | MV_S2_RiskUp: -6.02% | MV_S3_RiskMore: -5.16% | MV_S4_MinVar: -3.35% | Bench: -2.88%

============================================================
2024-11-08 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -8.53% | MV_S2_RiskUp: -7.32% | MV_S3_RiskMore: -5.69% | MV_S4_MinVar: -3.53% | Bench: -3.00%

============================================================
2024-11-15 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -8.03% | MV_S2_RiskUp: -7.57% | MV_S3_RiskMore: -4.59% | MV_S4_MinVar: -2.56% | Bench: -2.65%

============================================================
2024-11-22 | MEAN–VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -2.12% | MV_S2_RiskUp: -1.31% | MV_S3_RiskMore: -1.37% | MV_S4_MinVar: -0.38% | Bench: -0.08%
```

```
============================================================
2024-11-29 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.55% | MV_S2_RiskUp: 0.59% | MV_S3_RiskMore: 0.72% | MV_S4_MinVar: 0.67% | Bench: -0.69%

============================================================
2024-12-06 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 13.83% | MV_S2_RiskUp: 12.56% | MV_S3_RiskMore: 7.90% | MV_S4_MinVar: 3.81% | Bench: 2.41%

============================================================
2024-12-13 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -6.12% | MV_S2_RiskUp: -3.88% | MV_S3_RiskMore: -2.57% | MV_S4_MinVar: -0.96% | Bench: 0.79%

============================================================
2024-12-20 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: -5.95% | MV_S2_RiskUp: -5.39% | MV_S3_RiskMore: -5.05% | MV_S4_MinVar: -4.96% | Bench: -5.99%

============================================================
2024-12-27 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 1.79% | MV_S2_RiskUp: 1.15% | MV_S3_RiskMore: 1.31% | MV_S4_MinVar: 1.77% | Bench: 1.94%

============================================================
2025-01-03 | MEAN-VARIANCE (4 SKENARIO)
Return | MV_S1_Equal: 2.42% | MV_S2_RiskUp: 1.98% | MV_S3_RiskMore: 0.99% | MV_S4_MinVar: 0.86% | Bench: 0.61%

============================================================
BACKTEST MEAN-VARIANCE (4 SKENARIO) SELESAI
```

In [14]:

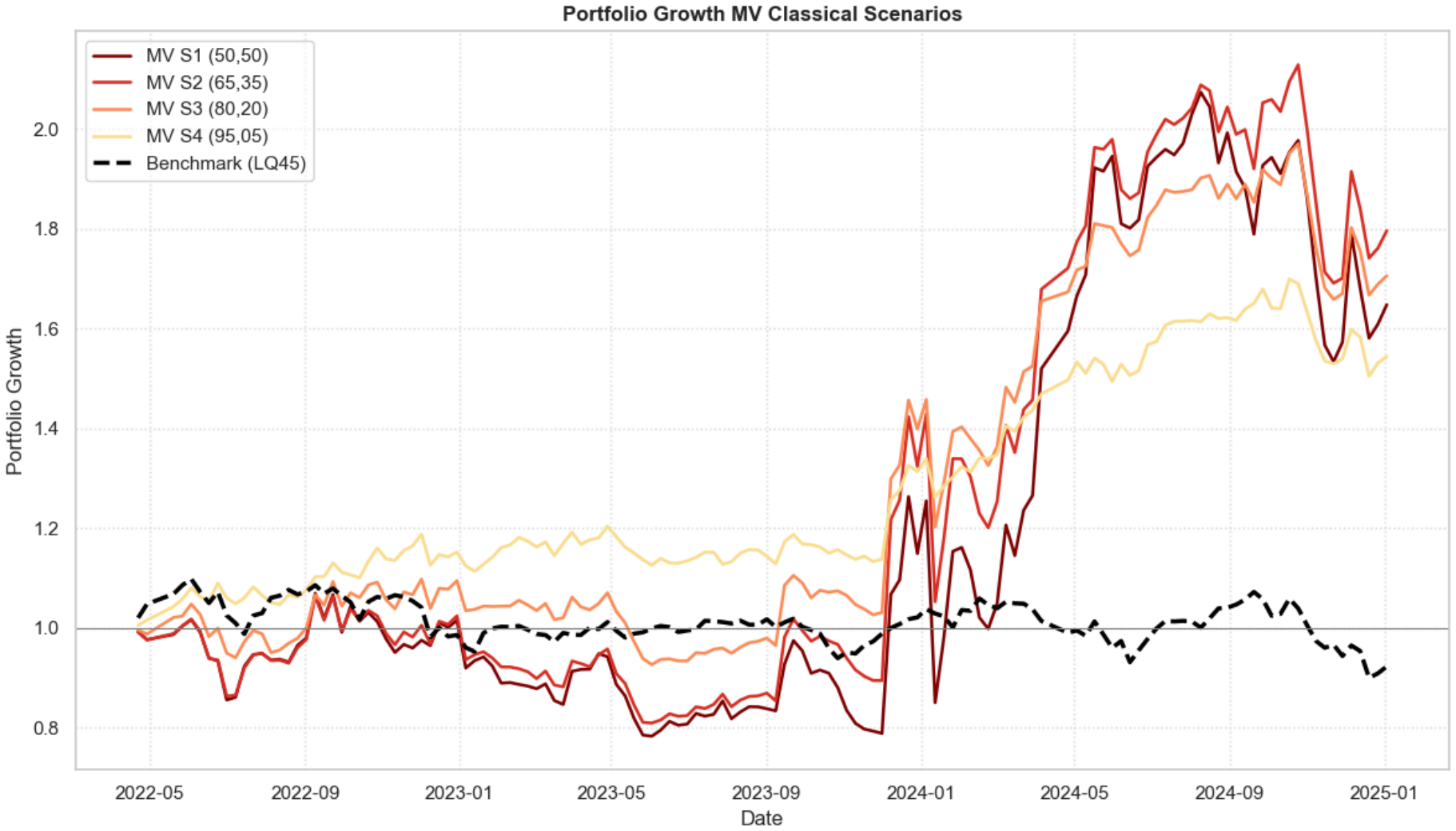
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# ========================================
# 1. PERSIAPAN DATA BENCHMARK LQ45
# ========================================
df_lq45 = pd.read_csv('JKLQ45_daily_2022_2025.csv')
df_lq45 = df_lq45.iloc[2:].copy()
df_lq45.columns = ['Date','Close','High','Low','Open','Volume']
df_lq45['Date'] = pd.to_datetime(df_lq45['Date'])
df_lq45['Close'] = pd.to_numeric(df_lq45['Close'])
df_lq45.set_index('Date', inplace=True)
```

```
lq45_weekly_ret = df_lq45['Close'].resample('W-FRI').last().ffill().pct_change().fillna(0)

dates_bt = pd.to_datetime(dates_backtest).tz_localize(None)
lq45_aligned = lq45_weekly_ret.reindex(dates_bt, method='ffill').fillna(0).values

min_len_mv = min(len(portfolio_return_mv["MV_S1_Equal"]), len(lq45_aligned))
dates_bt = dates_bt[-min_len_mv:]
lq45_bench = lq45_aligned[-min_len_mv:]

# ==========================================
# 2. RETURN PER SKENARIO (MV)
# ==========================================
scenarios_mv = {
    "MV S1 (50,50)": np.array(portfolio_return_mv["MV_S1_Equal"])[-min_len_mv:],
    "MV S2 (65,35)": np.array(portfolio_return_mv["MV_S2_RiskUp"])[-min_len_mv:],
    "MV S3 (80,20)": np.array(portfolio_return_mv["MV_S3_RiskMore"])[-min_len_mv:],
    "MV S4 (95,05)": np.array(portfolio_return_mv["MV_S4_MinVar"])[-min_len_mv:],
    "Benchmark (LQ45)": lq45_bench
}

# ==========================================
# 3. FUNGSI METRIK
# ==========================================
def get_performance_metrics(returns, benchmark, is_benchmark=False):
    ann_factor = 52
    excess_ret = returns - benchmark if not is_benchmark else returns

    sd_ann = np.std(excess_ret) * np.sqrt(ann_factor)

    if is_benchmark:
        ir_ann = np.nan
    else:
        er_ann = np.mean(excess_ret) * ann_factor
        ir_ann = er_ann / sd_ann if sd_ann != 0 else np.nan

    # Portfolio Growth (NAV)
    wealth_path = np.cumprod(1 + returns)
    portfolio_growth = wealth_path[-1]

    # Cumulative Return (%)
    cumulative_return = portfolio_growth - 1
```

```
    # Maximum Drawdown
    running_max = np.maximum.accumulate(wealth_path)
    md = ((wealth_path - running_max) / running_max).min()

    return [sd_ann, ir_ann, md, portfolio_growth, cumulative_return]

# ==========================================
# 4. TABEL METRIK
# ==========================================
metrics_mv_list = []
for name, ret in scenarios_mv.items():
    is_bench = "Benchmark" in name
    m = get_performance_metrics(ret, lq45_bench, is_benchmark=is_bench)

    metrics_mv_list.append({
        "Model Strategy": name,
        "SD (Risk)": round(m[0], 4),
        "IR (Sharpe)": round(m[1], 4) if not np.isnan(m[1]) else "-",
        "Max Drawdown": f"{round(m[2]*100,2)}%",
        "Portfolio Growth (NAV)": round(m[3], 4),
        "Cumulative Return (%)": f"{round(m[4]*100,2)}%"
    })

df_mv_final = pd.DataFrame(metrics_mv_list)
print("\n=== PERFORMANCE METRICS (ANNUALIZED | MV CONVENTIONAL) ===")
print(df_mv_final.to_string(index=False))

# ==========================================
# 5. VISUALISASI PORTFOLIO GROWTH
# ==========================================
plt.figure(figsize=(12,7))

red_shades = ['#7f0000', '#d73027', '#fc8d59', '#fee090']

idx = 0
for name, ret in scenarios_mv.items():
    cum = np.cumprod(1 + ret)
    if "Benchmark" in name:
        plt.plot(dates_bt, cum, label=name, color='black', linestyle='--', linewidth=2.5)
    else:
```

```
        plt.plot(dates_bt, cum, label=name, color=red_shades[idx % 4], linewidth=1.8)
        idx += 1

plt.title("Portfolio Growth MV Classical Scenarios", fontweight='bold')
plt.xlabel("Date")
plt.ylabel("Portfolio Growth")
plt.legend()
plt.grid(True, linestyle=':', alpha=0.6)
plt.axhline(1, color='grey', linewidth=0.8)

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

```
=== PERFORMANCE METRICS (ANNUALIZED | MV CONVENTIONAL) ===
  Model Strategy  SD (Risk) IR (Sharpe) Max Drawdown  Portfolio Growth (NAV) Cumulative Return (%)
   MV S1 (50,50)     0.4702      0.6833      -32.68%                  1.6476                64.76%
   MV S2 (65,35)     0.4093      0.7979      -26.32%                  1.7957                79.57%
   MV S3 (80,20)     0.2816      0.9388      -17.52%                  1.7055                70.55%
   MV S4 (95,05)     0.1250      1.5664      -11.47%                  1.5439                54.39%
Benchmark (LQ45)     0.1300           -      -18.01%                  0.9232                -7.68%
```



In [15]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# ========================================
# 0. PERSIAPAN DATA (BENCHMARK & PORTFOLIO)
# ========================================
# 1. Olah Benchmark LQ45
```

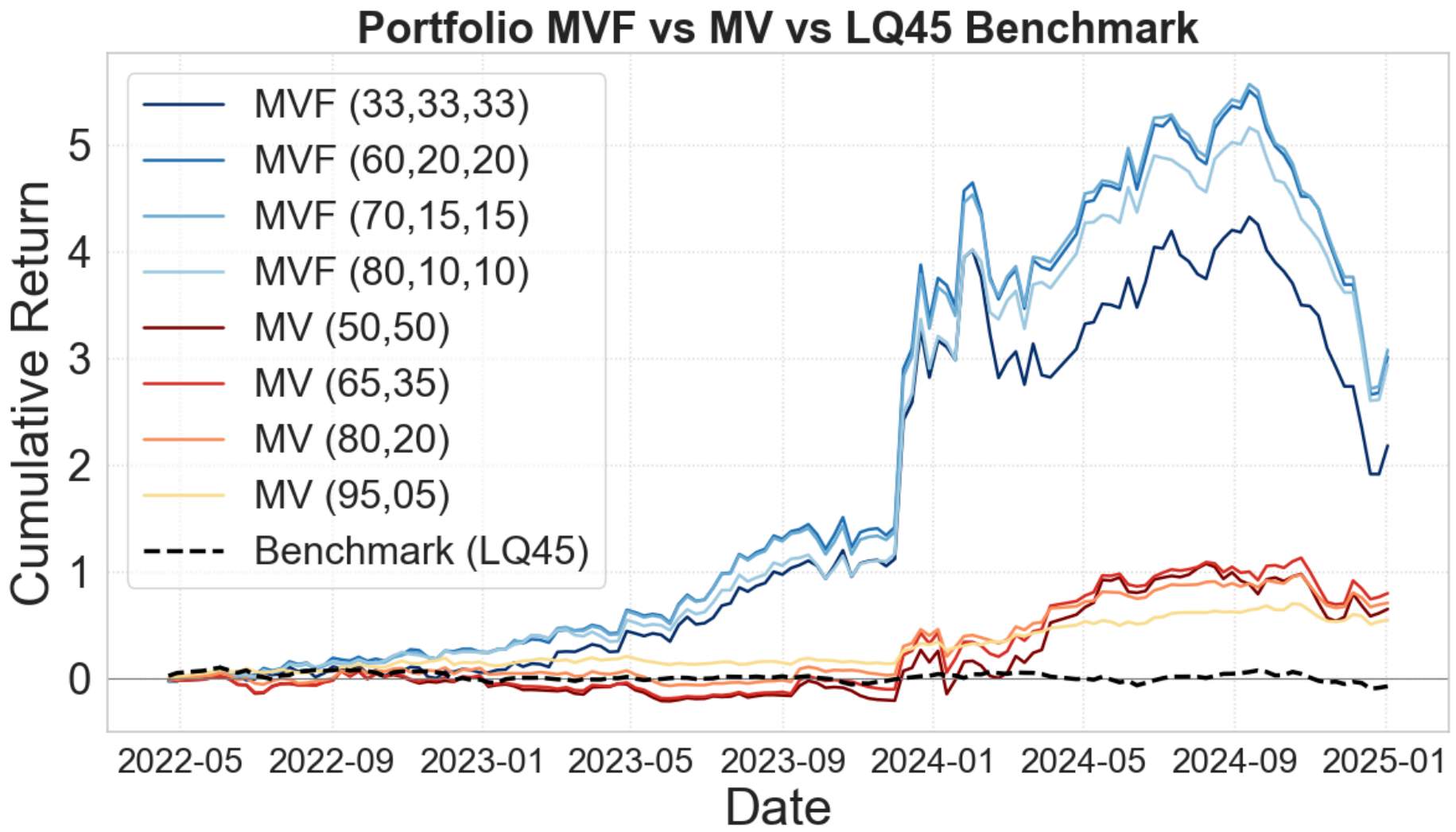
```
df_lq45 = pd.read_csv('JKLQ45_daily_2022_2025.csv')
df_lq45 = df_lq45.iloc[2:].copy()
df_lq45.columns = ['Date','Close','High','Low','Open','Volume']
df_lq45['Date'] = pd.to_datetime(df_lq45['Date'])
df_lq45['Close'] = pd.to_numeric(df_lq45['Close'])
df_lq45.set_index('Date', inplace=True)

lq45_weekly_price = df_lq45['Close'].resample('W-FRI').last().ffill()
lq45_weekly_return = lq45_weekly_price.pct_change().fillna(0)

# Hapus timezone agar sinkron
dates_bt = pd.to_datetime(dates_backtest).tz_localize(None)
lq45_ready = lq45_weekly_return.reindex(dates_bt, method='ffill').fillna(0).values

# 2. Samakan Panjang Data (Alignment)
min_len = min(
    len(portfolio_return_mvf["MVF_S1_Equal"]),
    len(portfolio_return_mvf["MVF_S2_RiskUp"]),
    len(portfolio_return_mvf["MVF_S3_RiskMore"]),
    len(portfolio_return_mvf["MVF_S4_VeryRisk"]),
    len(portfolio_return_mv["MV_S1_Equal"]),
    len(portfolio_return_mv["MV_S2_RiskUp"]),
    len(portfolio_return_mv["MV_S3_RiskMore"]),
    len(portfolio_return_mv["MV_S4_MinVar"]),
    len(lq45_ready)
)
dates_bt = dates_bt[-min_len:]

# 3. MASUKKAN SEMUA DATA KE returns_all (TERMASUK BENCHMARK)
returns_all = {
    "MVF (33,33,33)": np.array(portfolio_return_mvf["MVF_S1_Equal"])[-min_len:],
    "MVF (60,20,20)": np.array(portfolio_return_mvf["MVF_S2_RiskUp"])[-min_len:],
    "MVF (70,15,15)": np.array(portfolio_return_mvf["MVF_S3_RiskMore"])[-min_len:],
    "MVF (80,10,10)": np.array(portfolio_return_mvf["MVF_S4_VeryRisk"])[-min_len:],
    "MV (50,50)":      np.array(portfolio_return_mv["MV_S1_Equal"])[-min_len:],
    "MV (65,35)":      np.array(portfolio_return_mv["MV_S2_RiskUp"])[-min_len:],
    "MV (80,20)":      np.array(portfolio_return_mv["MV_S3_RiskMore"])[-min_len:],
    "MV (95,05)":      np.array(portfolio_return_mv["MV_S4_MinVar"])[-min_len:],
    "Benchmark (LQ45)":   lq45_ready[-min_len:]
}
```

```python
# ==========================================
# 1. FUNGSI METRIK (STANDAR HUANG ET AL. 2026)
# ==========================================
ann_factor = 52

def get_metrics(returns, benchmark, is_benchmark=False):
    # Jika menghitung benchmark itu sendiri, excess return adalah 0
    excess_ret = returns - benchmark

    # 1. Standard Deviation (Annualized) [cite: 305]
    # Untuk benchmark, kita hitung SD dari return aslinya
    sd_base = returns if is_benchmark else excess_ret
    sd_ann = np.std(sd_base) * np.sqrt(ann_factor)

    # 2. Sharpe / Information Ratio (Annualized) [cite: 308-309]
    # Catatan: IR benchmark terhadap dirinya sendiri akan 0, jadi kita beri tanda '-'
    if is_benchmark:
        ir_ann = np.nan
    else:
        er_ann = np.mean(excess_ret) * ann_factor
        ir_ann = er_ann / sd_ann if sd_ann != 0 else np.nan

    # 3. Total Return (Cumulative) [cite: 306]
    wealth_path = np.cumprod(1 + returns)
    tor = wealth_path[-1]

    # 4. Maximum Drawdown [cite: 312-313]
    running_max = np.maximum.accumulate(wealth_path)
    md = ((wealth_path - running_max) / running_max).min()

    return [sd_ann, ir_ann, md, tor]

# ==========================================
# 2. TABEL METRIK (TERMASUK BARIS BENCHMARK)
# ==========================================
final_metrics = []
bench_raw = returns_all["Benchmark (LQ45)"]

for name, ret in returns_all.items():
    is_bench = True if "Benchmark" in name else False
    m = get_metrics(ret, bench_raw, is_benchmark=is_bench)
```

```
    final_metrics.append({
        "Model Strategy": name,
        "SD (Risk)": round(m[0], 4),
        "IR (Sharpe)": round(m[1], 4) if not np.isnan(m[1]) else "-",
        "MaxDD": f"{round(m[2] * 100, 2)}%",
        "Total Return": round(m[3], 4)
    })

df_final = pd.DataFrame(final_metrics)
print("\n=== FINAL PERFORMANCE METRICS (COMPARED TO BENCHMARK) ===")
print(df_final.to_string(index=False))

# ==========================================
# 3. PLOT (NET VALUE)
# ==========================================
plt.figure(figsize=(12, 7))
# Skema warna sesuai sebelumnya
colors_mvf = ['#08306b', '#2171b5', '#6baed6', '#9ecae1']
colors_mv = ['#7f0000', '#d73027', '#fc8d59', '#fee090']

i_f, i_v = 0, 0
for name, ret in returns_all.items():
    cum = np.cumprod(1 + ret) - 1
    if "Benchmark" in name:
        plt.plot(dates_bt, cum, label=name, color='black', linestyle='--', linewidth=2.5, zorder=10)
    elif "MVF" in name:
        plt.plot(dates_bt, cum, label=name, color=colors_mvf[i_f % 4], linewidth=1.8)
        i_f += 1
    elif "MV" in name:
        plt.plot(dates_bt, cum, label=name, color=colors_mv[i_v % 4], linewidth=1.8)
        i_v += 1

plt.title("Portfolio MVF vs MV vs LQ45 Benchmark", fontweight='bold', fontsize=25)
plt.xlabel("Date", fontsize=30)
plt.ylabel("Cumulative Return", fontsize=30)
plt.legend(loc='best', frameon=True, fontsize=23)
plt.grid(True, linestyle=':', alpha=0.6)
plt.axhline(0, color='grey', linewidth=0.8)
plt.xticks(fontsize=20)
plt.yticks(fontsize=25)
```

```
for spine in plt.gca().spines.values(): spine.set_visible(True)
plt.tight_layout()
plt.show()
```

```
=== FINAL PERFORMANCE METRICS (COMPARED TO BENCHMARK) ===
  Model Strategy  SD (Risk) IR (Sharpe)   MaxDD  Total Return
  MVF (33,33,33)     0.5376      1.0947  -45.3%        3.1805
  MVF (60,20,20)     0.5221       1.279  -43.8%        4.0136
  MVF (70,15,15)     0.5150      1.3014 -43.46%        4.0784
  MVF (80,10,10)     0.5016       1.297 -41.57%        3.9426
      MV (50,50)     0.4702      0.6833 -32.68%        1.6476
      MV (65,35)     0.4093      0.7979 -26.32%        1.7957
      MV (80,20)     0.2816      0.9388 -17.52%        1.7055
      MV (95,05)     0.1250      1.5664 -11.47%        1.5439
Benchmark (LQ45)     0.1300           - -18.01%        0.9232
```



In [16]:
```
df_lq45 = pd.read_csv('JKLQ45_2015-2024_daily.csv')
df_lq45 = df_lq45.iloc[2:].copy()
df_lq45.columns = ['Date','Close','High','Low','Open','Volume']
df_lq45['Date'] = pd.to_datetime(df_lq45['Date'])
df_lq45['Close'] = pd.to_numeric(df_lq45['Close'])
df_lq45.set_index('Date', inplace=True)

# Weekly price
```

```
lq45_weekly_price = (
    df_lq45['Close']
    .resample('W-FRI')
    .last()
    .ffill()
)

# Weekly return (HMM-safe)
lq45_ret = lq45_weekly_price.pct_change().dropna()

lq45_ret.head()
```

Out[16]:
```
Date
2015-01-09   -0.004761
2015-01-16   -0.013896
2015-01-23    0.045332
2015-01-30   -0.015618
2015-02-06    0.012982
Freq: W-FRI, Name: Close, dtype: float64
```

In [18]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from hmmlearn.hmm import GaussianHMM

# ==========================
# LOAD DATA
# ==========================
df_lq45 = pd.read_csv('JKLQ45_2015-2024_daily.csv')
df_lq45 = df_lq45.iloc[2:].copy()
df_lq45.columns = ['Date', 'Close', 'High', 'Low', 'Open', 'Volume']

df_lq45['Date'] = pd.to_datetime(df_lq45['Date'])
df_lq45['Close'] = pd.to_numeric(df_lq45['Close'], errors='coerce')
df_lq45 = df_lq45.dropna(subset=['Date', 'Close'])
df_lq45 = df_lq45.sort_values('Date')
df_lq45.set_index('Date', inplace=True)

# ==========================
# WEEKLY DATA
# ==========================
```

```
lq45_weekly_price = df_lq45['Close'].resample('W-FRI').last().ffill()
lq45_ret = lq45_weekly_price.pct_change()

# ==========================
# PARAMETER
# ==========================
WINDOW = 235
VOL_WINDOW = 12
BACKTEST_START = "2022-04-22"

THRESH_BULL = 0.60
THRESH_BEAR = 0.60

# ==========================
# FEATURE
# ==========================
volatility = lq45_ret.rolling(VOL_WINDOW).std()

data = pd.concat([
    lq45_ret.rename("ret"),
    volatility.rename("vol")
], axis=1).dropna()

start_idx = data.index.get_loc(pd.Timestamp(BACKTEST_START))

# ==========================
# ROLLING HMM + PREDICTION
# ==========================
results = []
eval_results = []
prev_regime = None

for t in range(start_idx, len(data)):
    train = data.iloc[t-WINDOW:t]

    if len(train) < WINDOW:
        continue

    X = train[['ret', 'vol']].values

    mu = X.mean(axis=0)
```

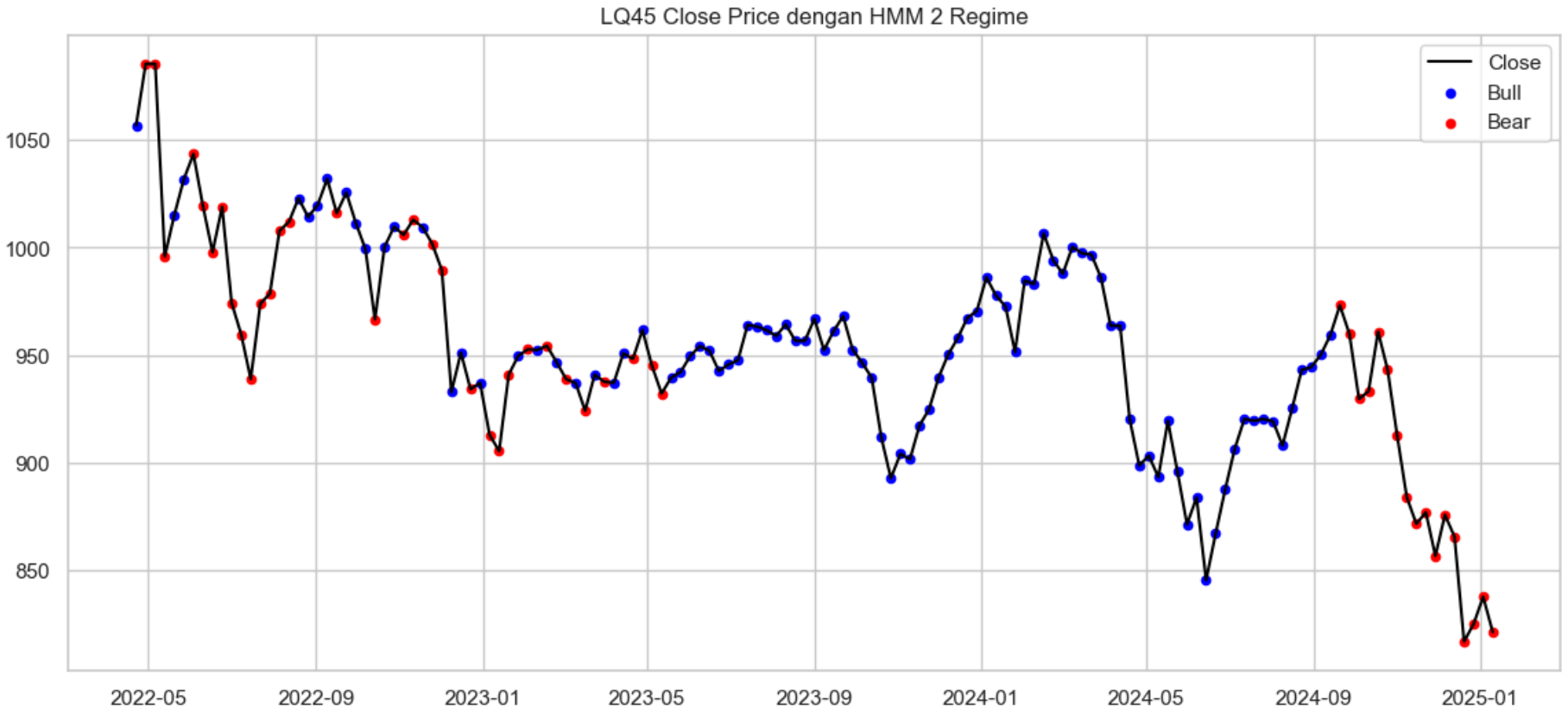
```
    sd = X.std(axis=0)

    if np.any(sd == 0) or np.any(np.isnan(sd)):
        continue

    X_scaled = (X - mu) / sd

    model = GaussianHMM(
        n_components=2,
        covariance_type='full',
        n_iter=150,
        random_state=46,
    )

    model.fit(X_scaled)

    probs = model.predict_proba(X_scaled)
    last_probs = probs[-1]

    means_return = model.means_[:,0]
    vols = np.sqrt(model.covars_[:,1,1])
    if means_return[0] > means_return[1]:
        bull_state = 0
        bear_state = 1
    else:
        bull_state = 1
        bear_state = 0

    next_probs = last_probs @ model.transmat_

    p_bull = next_probs[bull_state]
    p_bear = next_probs[bear_state]

    if prev_regime is None:
        regime = 'bull' if p_bull >= p_bear else 'bear'
    else:
        if prev_regime == 'bull':
            regime = 'bear' if p_bear > THRESH_BEAR else 'bull'
        else:
            regime = 'bull' if p_bull > THRESH_BULL else 'bear'
```

```
    results.append({
        'Date': data.index[t],
        'close': lq45_weekly_price.loc[data.index[t]],
        'p_bull': p_bull,
        'p_bear': p_bear,
        'regime': regime
    })

    logL = model.score(X_scaled)

    n = X_scaled.shape[0]
    k = X_scaled.shape[1]
    n_states = model.n_components

    n_params = (
        (n_states - 1) +
        n_states * (n_states - 1) +
        n_states * k +
        n_states * k * (k + 1) / 2
    )

    aic = -2 * logL + 2 * n_params
    bic = -2 * logL + np.log(n) * n_params

    transmat = model.transmat_

    eval_results.append({
        'Date': data.index[t],
        'logL': logL,
        'AIC': aic,
        'BIC': bic,
        'bull_mean_ret': means_return[bull_state],
        'bear_mean_ret': means_return[bear_state],
        'bull_vol': vols[bull_state],
        'bear_vol': vols[bear_state],
        'bull_to_bull': transmat[bull_state, bull_state],
        'bull_to_bear': transmat[bull_state, bear_state],
        'bear_to_bull': transmat[bear_state, bull_state],
        'bear_to_bear': transmat[bear_state, bear_state]
    })
```

```
    prev_regime = regime

hasil = pd.DataFrame(results).set_index('Date')
eval_df = pd.DataFrame(eval_results).set_index('Date')

# ========================
# DURASI REGIME
# ========================
hasil['regime_change'] = hasil['regime'] != hasil['regime'].shift(1)
hasil['regime_group'] = hasil['regime_change'].cumsum()

duration_df = (
    hasil.groupby(['regime_group', 'regime'])
    .size()
    .reset_index(name='duration')
)

# ========================
# OUTPUT EVALUASI MODEL
# ========================
print("===== RATA-RATA EVALUASI MODEL =====")
print(eval_df[['logL', 'AIC', 'BIC']].mean())
print()

print("===== RATA-RATA TRANSITION PROBABILITY =====")
print(eval_df[['bull_to_bull', 'bull_to_bear', 'bear_to_bull', 'bear_to_bear']].mean())
print()

print("===== RATA-RATA MEAN RETURN STATE =====")
print(eval_df[['bull_mean_ret', 'bear_mean_ret']].mean())
print()

print("===== RATA-RATA MEAN Volatility STATE =====")
print(eval_df[['bull_vol', 'bear_vol']].mean())
print()

print("===== RATA-RATA DURASI REGIME =====")
print(duration_df.groupby('regime')['duration'].mean())
print()

print("===== RINGKASAN EVALUASI MODEL =====")
```

```
print(f"Rata-rata LogL       : {eval_df['logL'].mean():.4f}")
print(f"Rata-rata AIC        : {eval_df['AIC'].mean():.4f}")
print(f"Rata-rata BIC        : {eval_df['BIC'].mean():.4f}")
print(f"Avg Bull->Bull       : {eval_df['bull_to_bull'].mean():.4f}")
print(f"Avg Bear->Bear       : {eval_df['bear_to_bear'].mean():.4f}")
print(f"Avg Bull Mean Return : {eval_df['bull_mean_ret'].mean():.6f}")
print(f"Avg Bear Mean Return : {eval_df['bear_mean_ret'].mean():.6f}")

# ==========================
# PLOT 1: HARGA + REGIME
# ==========================
plt.figure(figsize=(14, 6))
plt.plot(hasil.index, hasil['close'], color='black', label='Close')

bull = hasil['regime'] == 'bull'
bear = hasil['regime'] == 'bear'

plt.scatter(hasil.index[bull], hasil.loc[bull, 'close'], color='blue', s=20, label='Bull')
plt.scatter(hasil.index[bear], hasil.loc[bear, 'close'], color='red', s=20, label='Bear')

plt.title('LQ45 Close Price dengan HMM 2 Regime')
plt.legend()
plt.grid(True)
plt.show()
```

```
===== RATA-RATA EVALUASI MODEL =====
logL    -491.591359
AIC     1009.182719
BIC     1054.157331
dtype: float64

===== RATA-RATA TRANSITION PROBABILITY =====
bull_to_bull    0.968497
bull_to_bear    0.031503
bear_to_bull    0.048167
bear_to_bear    0.951833
dtype: float64

===== RATA-RATA MEAN RETURN STATE =====
bull_mean_ret    0.061865
bear_mean_ret   -0.182474
dtype: float64

===== RATA-RATA MEAN Volatility STATE =====
bull_vol    0.481799
bear_vol    0.846578
dtype: float64

===== RATA-RATA DURASI REGIME =====
regime
bear    3.0625
bull    5.8750
Name: duration, dtype: float64

===== RINGKASAN EVALUASI MODEL =====
Rata-rata LogL       : -491.5914
Rata-rata AIC        : 1009.1827
Rata-rata BIC        : 1054.1573
Avg Bull->Bull       : 0.9685
Avg Bear->Bear       : 0.9518
Avg Bull Mean Return : 0.061865
Avg Bear Mean Return : -0.182474
```



In [19]:
```
eval_df.head()
```

Out[19]:

Date	logL	AIC	BIC	bull_mean_ret	bear_mean_ret	bull_vol	bear_vol	bull_to_bull	bull_to_bear	be
2022-04-22	-496.587404	1019.174808	1064.149420	0.019050	-0.300561	0.551609	0.776718	0.995453	0.004547	
2022-04-29	-503.102112	1032.204225	1077.178836	0.060323	-0.003246	0.322566	0.575153	0.916667	0.083333	
2022-05-06	-503.189131	1032.378262	1077.352874	0.056524	-0.003042	0.322559	0.575173	0.916667	0.083333	
2022-05-13	-503.282295	1032.564591	1077.539202	0.059521	-0.003203	0.322444	0.575498	0.916667	0.083333	
2022-05-20	-499.149500	1024.299000	1069.273612	0.018269	-0.288240	0.551229	0.778018	0.995454	0.004546	

In [20]:

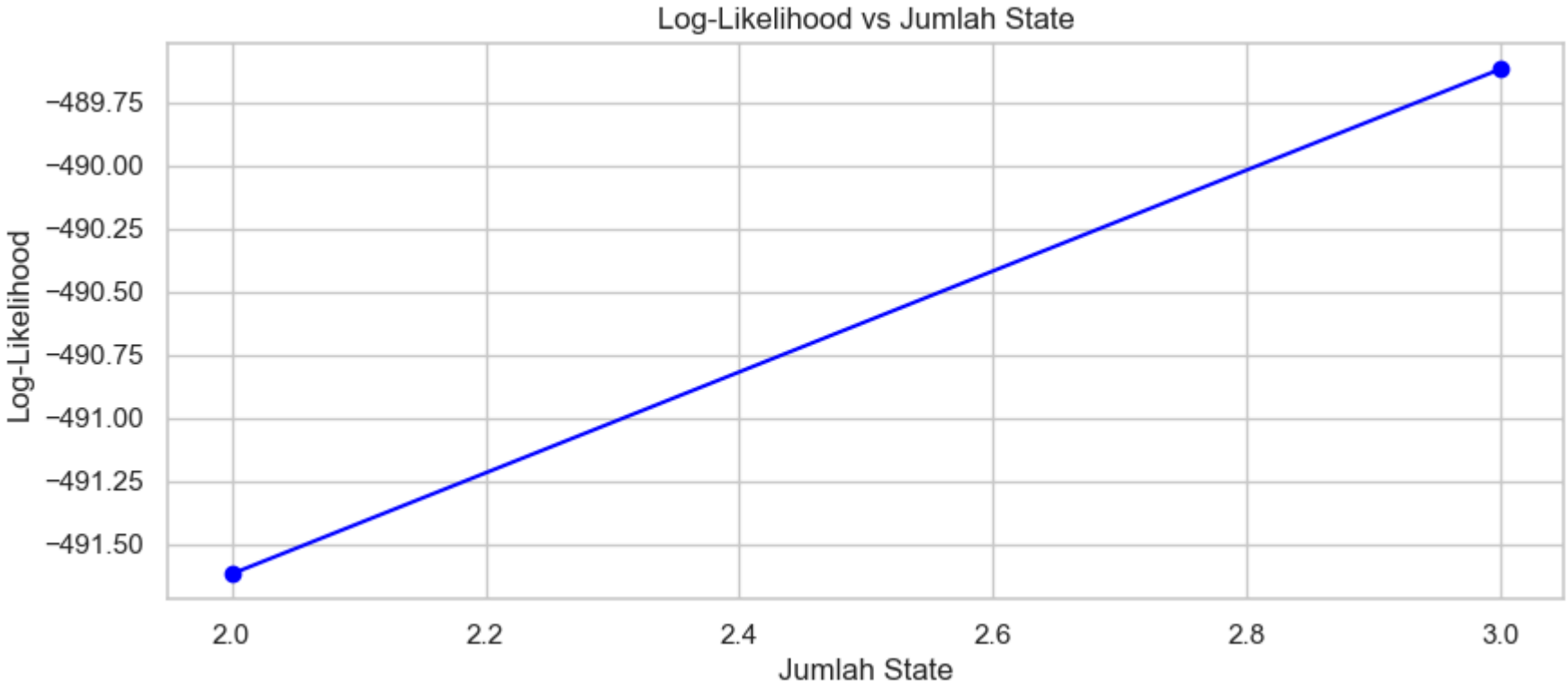
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from hmmlearn.hmm import GaussianHMM

# ==========================
# LOAD DATA
# ==========================
df_lq45 = pd.read_csv('JKLQ45_2015-2024_daily.csv')
df_lq45 = df_lq45.iloc[2:].copy()
df_lq45.columns = ['Date', 'Close', 'High', 'Low', 'Open', 'Volume']

df_lq45['Date'] = pd.to_datetime(df_lq45['Date'])
df_lq45['Close'] = pd.to_numeric(df_lq45['Close'], errors='coerce')
df_lq45 = df_lq45.dropna(subset=['Date', 'Close'])
df_lq45 = df_lq45.sort_values('Date')
df_lq45.set_index('Date', inplace=True)
```

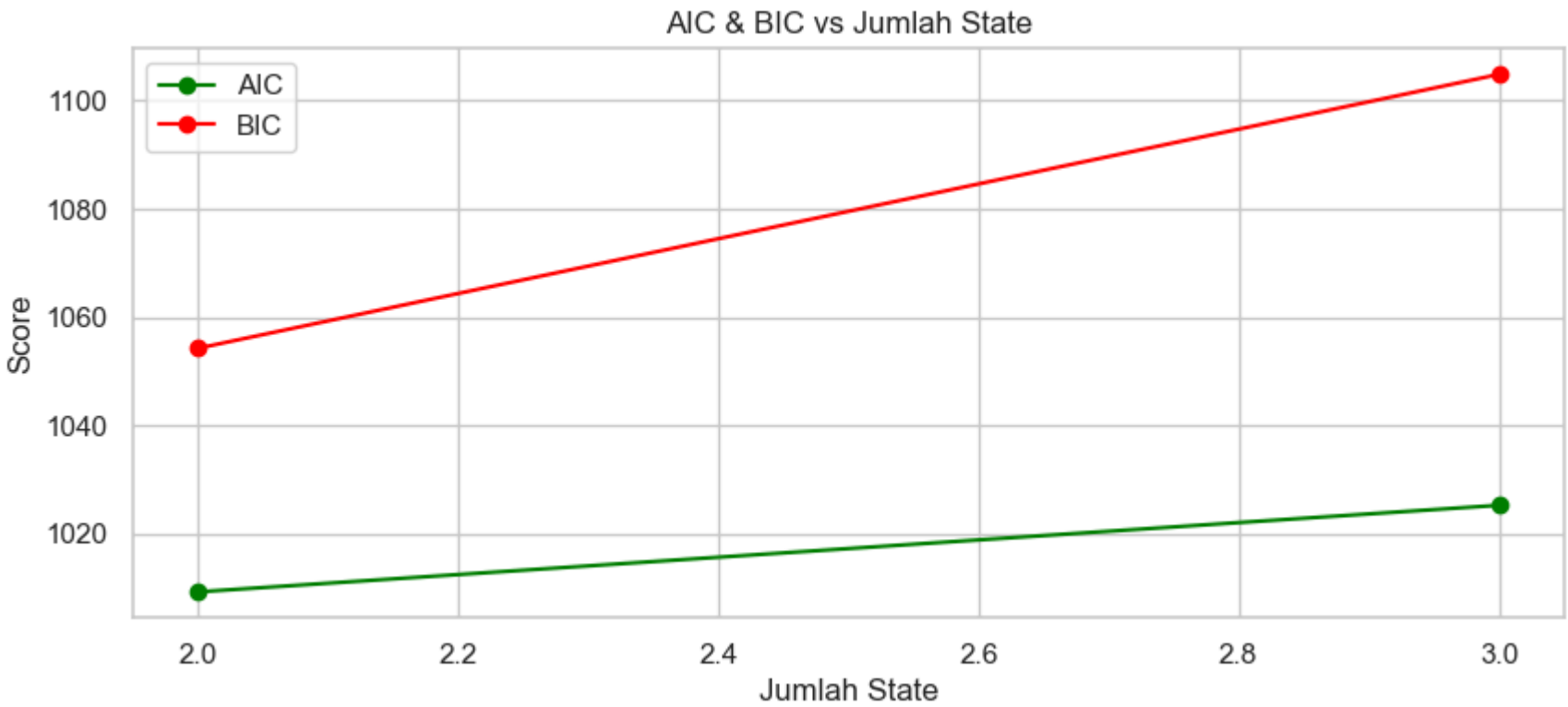
```
# ==========================
# WEEKLY DATA
# ==========================
lq45_weekly_price = df_lq45['Close'].resample('W-FRI').last().ffill()
lq45_ret = lq45_weekly_price.pct_change()

# ==========================
# PARAMETER
# ==========================
WINDOW = 235  # 235 weeks as the training window
BACKTEST_START = "2022-04-22"

# ==========================
# FEATURE
# ==========================
volatility = lq45_ret.rolling(12).std()  # 12 weeks for volatility calculation

data = pd.concat([
    lq45_ret.rename("ret"),
    volatility.rename("vol")
], axis=1).dropna()

# Use only the first 235 weeks of data for training
train = data.iloc[:WINDOW]

states_list = [2, 3]  # Using 2 and 3 states for evaluation
results_all = []

for n_state in states_list:
    # Use the first 235 weeks for training
    X = train[['ret', 'vol']].values  # Using log returns and volatility

    # Normalization of the data
    mu = X.mean(axis=0)
    sd = X.std(axis=0)

    if np.any(sd == 0) or np.any(np.isnan(sd)):
        continue

    X_scaled = (X - mu) / sd
```

```
    # Fit the HMM model
    model = GaussianHMM(
        n_components=n_state,
        covariance_type='full',
        n_iter=150,
        random_state=46
    )
    model.fit(X_scaled)

    logL = model.score(X_scaled)  # Log-likelihood of the model
    n = X_scaled.shape[0]  # Number of samples
    k = X_scaled.shape[1]  # Number of features (2 features: return and volatility)

    # Calculate AIC and BIC
    n_params = (n_state - 1) + n_state * (n_state - 1) + n_state * k + n_state * k * (k + 1) / 2
    aic = -2 * logL + 2 * n_params
    bic = -2 * logL + np.log(n) * n_params

    results_all.append({
        'n_state': n_state,
        'logL': logL,
        'AIC': aic,
        'BIC': bic
    })

# Convert results into DataFrame
df_results = pd.DataFrame(results_all)
print(df_results)

# ---------------------------
# PLOT 1: Log-likelihood
# ---------------------------
plt.figure(figsize=(10, 4))
plt.plot(df_results['n_state'], df_results['logL'], marker='o', color='blue')
plt.title('Log-Likelihood vs Jumlah State')
plt.xlabel('Jumlah State')
plt.ylabel('Log-Likelihood')
plt.grid(True)
plt.show()

# ---------------------------
```

```
# PLOT 2: AIC & BIC
# --------------------------
plt.figure(figsize=(10, 4))
plt.plot(df_results['n_state'], df_results['AIC'], marker='o', label='AIC', color='green')
plt.plot(df_results['n_state'], df_results['BIC'], marker='o', label='BIC', color='red')
plt.title('AIC & BIC vs Jumlah State')
plt.xlabel('Jumlah State')
plt.ylabel('Score')
plt.legend()
plt.grid(True)
plt.show()
```

```
   n_state        logL          AIC          BIC
0        2 -491.619536  1009.239072  1054.213684
1        3 -489.617410  1025.234820  1104.805287
```





In [21]:
```
rf_rate_annual = 0.05
rf_rate_weekly = np.exp(rf_rate_annual/52) - 1
df_regime = hasil[['regime']].copy()
df_regime['Bear_Market'] = (df_regime['regime'] == 'bear').astype(int)
df_regime = df_regime[['Bear_Market']]
df_regime.index.name = 'Date'
df_regime = df_regime.reset_index()

df_regime.head()
```

Out[21]:

	Date	Bear_Market
0	2022-04-22	0
1	2022-04-29	1
2	2022-05-06	1
3	2022-05-13	1
4	2022-05-20	0

In [22]:
```
ret_mvf_equity = np.array(portfolio_return_mvf["MVF_S3_RiskMore"])

df_mvf = pd.DataFrame({
    "Date": dates_bt,
    "MVF_Return": ret_mvf_equity
})
print(df_mvf)
```

```
          Date    MVF_Return
0   2022-04-22 -1.714174e-02
1   2022-04-29  5.306082e-08
2   2022-05-20  5.065353e-02
3   2022-05-27 -1.510570e-02
4   2022-06-03  7.396448e-02
..         ...           ...
133 2024-12-06  2.573274e-08
134 2024-12-13 -1.037464e-01
135 2024-12-20 -1.297297e-01
136 2024-12-27  5.579052e-03
137 2025-01-03  9.150323e-02

[138 rows x 2 columns]
```

In [23]:
```
df_merge = pd.merge(df_mvf, df_regime, on="Date", how="inner")

print(df_merge)
print(len(df_merge))
```

```
          Date    MVF_Return  Bear_Market
0   2022-04-22 -1.714174e-02            0
1   2022-04-29  5.306082e-08            1
2   2022-05-20  5.065353e-02            0
3   2022-05-27 -1.510570e-02            0
4   2022-06-03  7.396448e-02            1
..         ...           ...          ...
133 2024-12-06  2.573274e-08            1
134 2024-12-13 -1.037464e-01            1
135 2024-12-20 -1.297297e-01            1
136 2024-12-27  5.579052e-03            1
137 2025-01-03  9.150323e-02            1

[138 rows x 3 columns]
138
```

In [24]:

```python
import numpy as np

# ======================================================
# 1. DATA DARI MERGE
# ======================================================
ret_mvf_equity = df_merge["MVF_Return"].values
regime_flag = df_merge["Bear_Market"].values

T = len(df_merge)

# ======================================================
# 2. STRATEGI MVF + REGIME
# ======================================================
final_return_95 = []

for _, row in df_merge.iterrows():

    if row["Bear_Market"]:   # BEAR
        w_rf = 0.95  # 78.69%
        w_eq = 0.05 # Sisanya (100% - 78.69%)
    else:                    # BULL
        w_rf = 0.05  # 19.91%
        w_eq = 0.95  # Sisanya (100% - 19.91%)

    rp = w_eq * row["MVF_Return"] + w_rf * rf_rate_weekly
```

```
    final_return_95.append(rp)

final_return_95 = np.array(final_return_95)

# ======================================================
# 3. WEALTH
# ======================================================
wealth_mvf = np.cumprod(1 + ret_mvf_equity)
wealth_regime_95 = np.cumprod(1 + final_return_95)

# ======================================================
# 4. CEK HASIL
# ======================================================
print("Periode dibandingkan :", T)
print("Final Wealth MVF     :", wealth_mvf[-1])
print("Final Wealth Regime  :", wealth_regime_95[-1])
print("Jumlah BEAR          :", np.sum(regime_flag))
```

```
Periode dibandingkan : 138
Final Wealth MVF     : 4.078413240651926
Final Wealth Regime  : 5.024094002769601
Jumlah BEAR          : 46
```

In [25]:

```
import numpy as np

# ======================================================
# 1. DATA DARI MERGE
# ======================================================
ret_mvf_equity = df_merge["MVF_Return"].values
regime_flag = df_merge["Bear_Market"].values

T = len(df_merge)

# ======================================================
# 2. STRATEGI MVF + REGIME
# ======================================================
final_return_85 = []

for _, row in df_merge.iterrows():

    if row["Bear_Market"]:   # BEAR
```

```
        w_rf = 0.85  # 78.69%
        w_eq = 0.15  # Sisanya (100% - 78.69%)
    else:                    # BULL
        w_rf = 0.15  # 19.91%
        w_eq = 0.85  # Sisanya (100% - 19.91%)

    rp = w_eq * row["MVF_Return"] + w_rf * rf_rate_weekly
    final_return_85.append(rp)

final_return_85 = np.array(final_return_85)

# ======================================================
# 3. WEALTH
# ======================================================
wealth_mvf = np.cumprod(1 + ret_mvf_equity)
wealth_regime_85 = np.cumprod(1 + final_return_85)

# ======================================================
# 4. CEK HASIL
# ======================================================
print("Periode dibandingkan :", T)
print("Final Wealth MVF     :", wealth_mvf[-1])
print("Final Wealth Regime  :", wealth_regime_85[-1])
print("Jumlah BEAR          :", np.sum(regime_flag))
```

```
Periode dibandingkan : 138
Final Wealth MVF     : 4.078413240651926
Final Wealth Regime  : 4.269814460725747
Jumlah BEAR          : 46
```

In [26]:

```
import numpy as np

# ======================================================
# 1. DATA DARI MERGE
# ======================================================
ret_mvf_equity = df_merge["MVF_Return"].values
regime_flag = df_merge["Bear_Market"].values

T = len(df_merge)

# ======================================================
```

```
# 2. STRATEGI MVF + REGIME
# ==========================================================
final_return_79 = []

for _, row in df_merge.iterrows():

    if row["Bear_Market"]:   # BEAR
        w_rf = 0.79  # 78.69%
        w_eq = 0.21  # Sisanya (100% - 78.69%)
    else:                    # BULL
        w_rf = 0.21  # 19.91%
        w_eq = 0.79  # Sisanya (100% - 19.91%)

    rp = w_eq * row["MVF_Return"] + w_rf * rf_rate_weekly
    final_return_79.append(rp)

final_return_79 = np.array(final_return_79)

# ==========================================================
# 3. WEALTH
# ==========================================================
wealth_mvf = np.cumprod(1 + ret_mvf_equity)
wealth_regime_79 = np.cumprod(1 + final_return_79)

# ==========================================================
# 4. CEK HASIL
# ==========================================================
print("Periode dibandingkan :", T)
print("Final Wealth MVF     :", wealth_mvf[-1])
print("Final Wealth Regime  :", wealth_regime_79[-1])
print("Jumlah BEAR          :", np.sum(regime_flag))
```

```
Periode dibandingkan : 138
Final Wealth MVF     : 4.078413240651926
Final Wealth Regime  : 3.8634885796076515
Jumlah BEAR          : 46
```

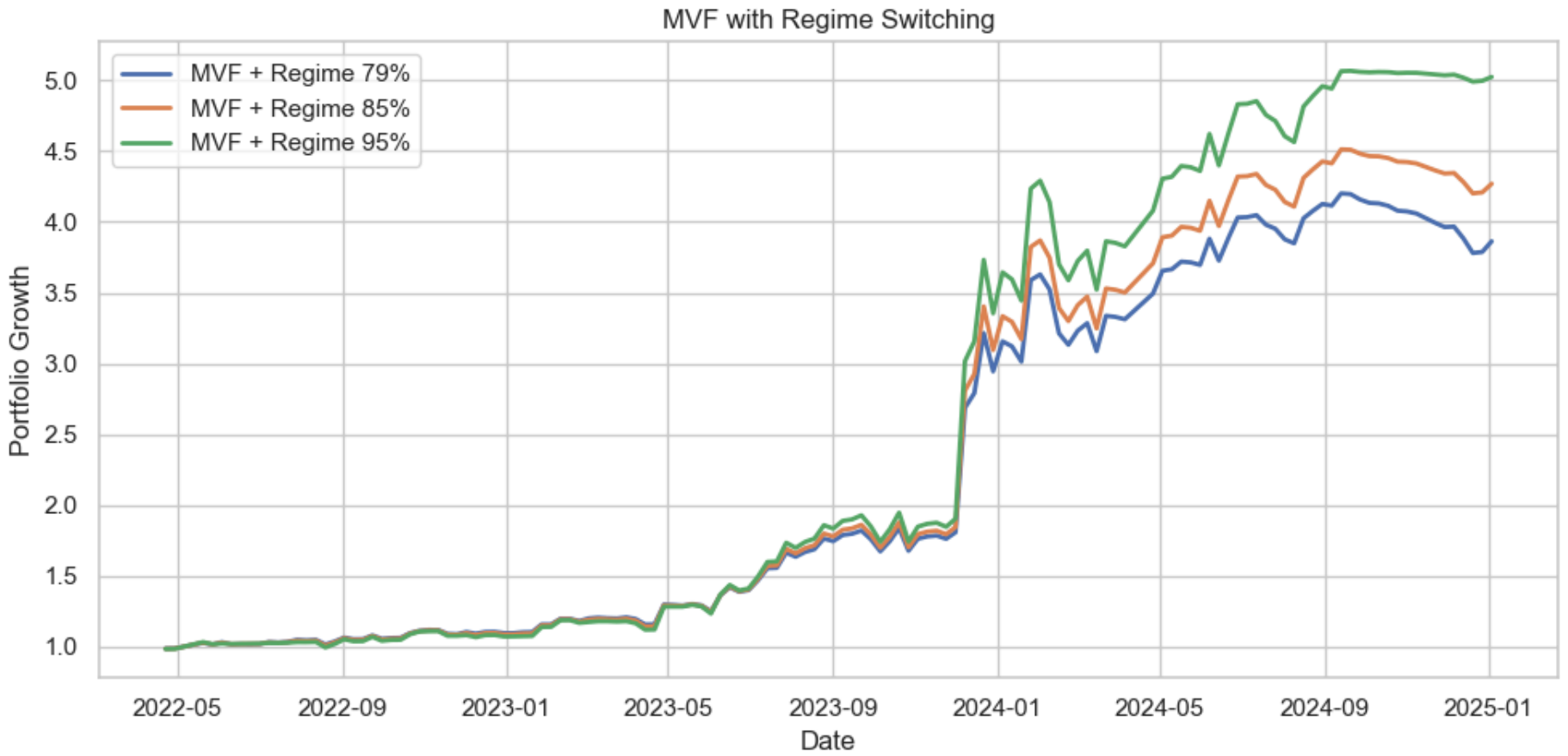
In [27]:
```
import pandas as pd
import numpy as np

# ==========================================
```

```
# 1. RETURN STRATEGY
# ==========================================
ret_mvf = ret_mvf_equity
ret_regime_79 = final_return_79
ret_regime_85 = final_return_85
ret_regime_95 = final_return_95

scenarios = {
    "MVF Only": ret_mvf,
    "MVF + Regime_79": ret_regime_79,
    "MVF + Regime_85": ret_regime_85,
    "MVF + Regime_95": ret_regime_95,
    "Benchmark (LQ45)": lq45_ready[-len(ret_mvf):]
}

# ==========================================
# 2. FUNGSI METRIK
# ==========================================
def get_metrics(returns, benchmark, is_benchmark=False):

    ann_factor = 52

    if is_benchmark:
        excess = returns
    else:
        excess = returns - benchmark

    # SD (Risk)
    sd = np.std(excess) * np.sqrt(ann_factor)

    # IR / Sharpe
    if is_benchmark:
        ir = np.nan
    else:
        er = np.mean(excess) * ann_factor
        ir = er / sd if sd != 0 else np.nan

    # Wealth path
    wealth = np.cumprod(1 + returns)

    # Portfolio Growth (NAV)
```

```
    nav = wealth[-1]

    # Cumulative Return
    cum_return = nav - 1

    # Max Drawdown
    running_max = np.maximum.accumulate(wealth)
    drawdown = ((wealth - running_max) / running_max).min()

    return [sd, ir, drawdown, nav, cum_return]

# ==========================================
# 3. BUAT TABEL
# ==========================================
metrics_list = []

for name, ret in scenarios.items():

    is_bench = "Benchmark" in name

    m = get_metrics(
        ret,
        scenarios["Benchmark (LQ45)"],
        is_benchmark=is_bench
    )

    metrics_list.append({
        "Model Strategy": name,
        "SD (Risk)": round(m[0],4),
        "IR (Sharpe)": round(m[1],4) if not np.isnan(m[1]) else "-",
        "Max Drawdown": f"{round(m[2]*100,2)}%",
        "Portfolio Growth (NAV)": round(m[3],4),
        "Cumulative Return (%)": f"{round(m[4]*100,2)}%"
    })

df_metrics = pd.DataFrame(metrics_list)

print("\n=== PERFORMANCE METRICS (REGIME STRATEGY) ===")
print(df_metrics.to_string(index=False))
```

```
=== PERFORMANCE METRICS (REGIME STRATEGY) ===
  Model Strategy  SD (Risk) IR (Sharpe) Max Drawdown  Portfolio Growth (NAV) Cumulative Return (%)
        MVF Only     0.5150      1.3014      -43.46%                  4.0784                307.84%
 MVF + Regime_79     0.3943      1.5181      -14.92%                  3.8635                286.35%
 MVF + Regime_85     0.4223      1.5288      -16.05%                  4.2698                326.98%
 MVF + Regime_95     0.4699       1.541      -17.92%                  5.0241                402.41%
Benchmark (LQ45)     0.1300           -      -18.01%                  0.9232                 -7.68%
```

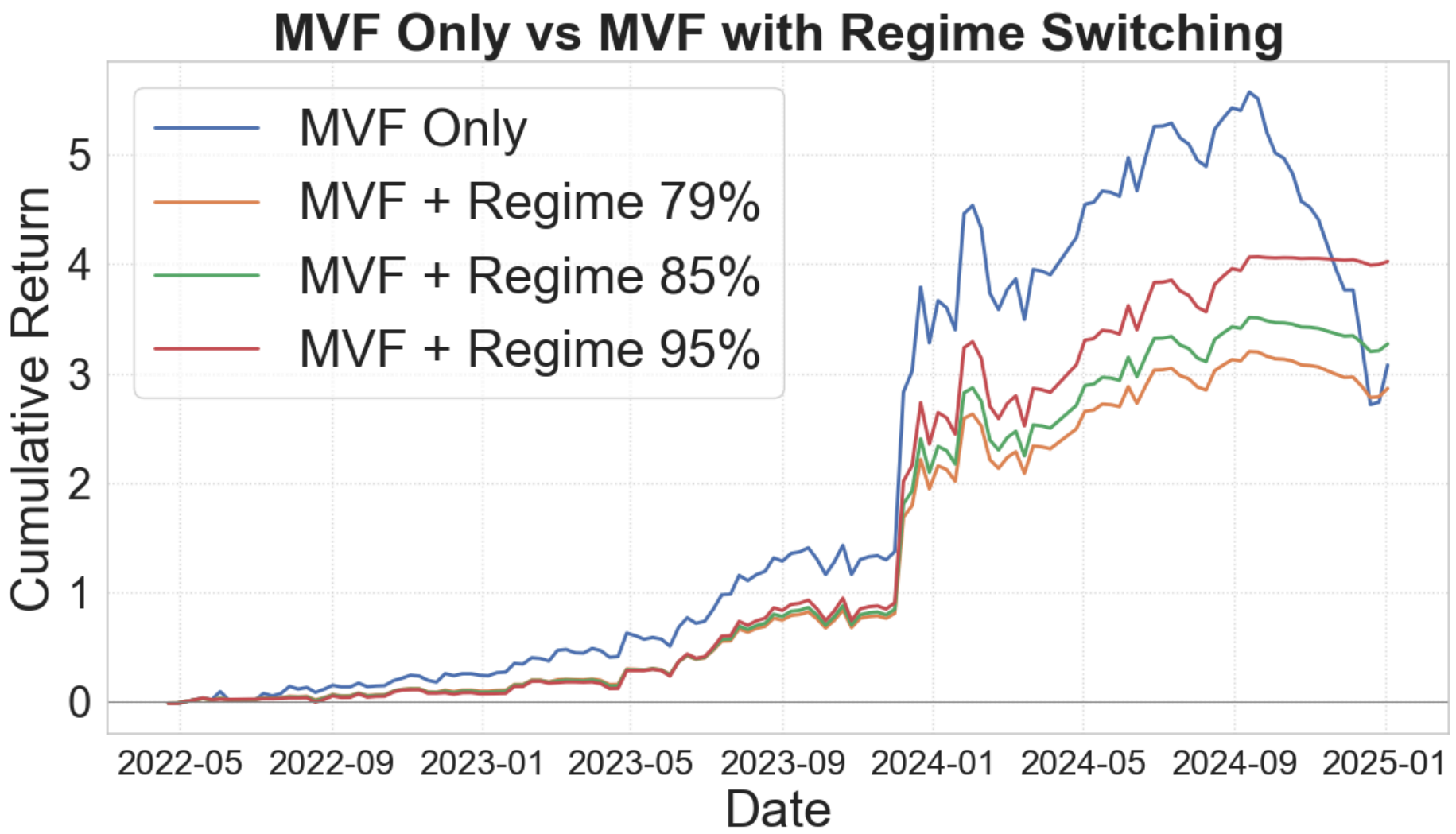
In [28]:
```python
import matplotlib.pyplot as plt

dates_plot = dates_bt[-len(wealth_mvf):]

plt.figure(figsize=(10,5))

#plt.plot(dates_plot, wealth_mvf, label="MVF Only", linewidth=2)
plt.plot(dates_plot, wealth_regime_79, label="MVF + Regime 79%", linewidth=2)
plt.plot(dates_plot, wealth_regime_85, label="MVF + Regime 85%", linewidth=2)
plt.plot(dates_plot, wealth_regime_95, label="MVF + Regime 95%", linewidth=2)



plt.xlabel("Date")
plt.ylabel("Portfolio Growth")
plt.title("MVF with Regime Switching")

plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



odel ini belajar dari pola masa lalu untuk mengelompokkan data ke dalam dua kategori (distribusi Gaussian) yang berbeda: satu kelompok dengan pergerakan stabil dan satu lagi dengan gejolak tinggi. Di setiap langkah waktu, model akan mencocokkan data terbaru dengan kedua kelompok tersebut, lalu memberikan label "Bearish" jika data saat ini lebih mirip dengan karakter kelompok yang memiliki volatilitas paling liar.

In [29]:
```
import matplotlib.pyplot as plt

dates_plot = dates_bt[-len(wealth_mvf):]

plt.figure(figsize=(12, 7))
cum_mvf = wealth_mvf - 1
cum_regime_79 = wealth_regime_79 - 1
```

```
cum_regime_85 = wealth_regime_85 - 1
cum_regime_95 = wealth_regime_95 - 1


plt.plot(dates_plot, cum_mvf, label="MVF Only", linewidth=2)
plt.plot(dates_plot, cum_regime_79, label="MVF + Regime 79%", linewidth=2)
plt.plot(dates_plot, cum_regime_85, label="MVF + Regime 85%", linewidth=2)
plt.plot(dates_plot, cum_regime_95, label="MVF + Regime 95%", linewidth=2)

plt.xlabel("Date", fontsize=30)
plt.ylabel("Cumulative Return", fontsize=30)
plt.title("MVF Only vs MVF with Regime Switching", fontweight='bold', fontsize=30)
plt.xticks(fontsize=20)
plt.yticks(fontsize=25)

plt.legend(loc='best', frameon=True, fontsize=30)
plt.grid(True, linestyle=':', alpha=0.6)
plt.axhline(0, color='grey', linewidth=0.8)
for spine in plt.gca().spines.values(): spine.set_visible(True)
plt.tight_layout()
plt.show()
```


MVF Only vs MVF with Regime Switching
MVF Only
MVF + Regime 79%
MVF + Regime 85%
MVF + Regime 95%
Cumulative Return
0
1
2
3
4
5
2022-05
2022-09
2023-01
2023-05
2023-09
2024-01
2024-05
2024-09
2025-01
Date