In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
from scipy.signal import savgol_filter
import warnings
warnings.filterwarnings('ignore')

# Reproducibility
np.random.seed(42)  # The answer to life, the universe, and 4:55pm emergencies

# Date index: 36 months
dates = pd.date_range(start='2022-01-01', periods=36, freq='MS')

# Trend: slow, inexorable, deeply demoralising upward drift
trend = np.linspace(4.0, 7.5, 36)

# Seasonality: 12-month cycle reflecting known aggravating factors
seasonal_pattern = np.array([
    2.5,   # January  - renewal season. chaos.
    0.5,   # February - brief respite. enjoy it.
    1.2,   # March    - Q1 close looms
    0.3,   # April    - calm. suspicious.
    0.6,   # May      - bank holidays everywhere. the emergency multiplies.
    1.1,   # June     - half-year targets
    2.8,   # July     - summer holidays begin. colleague cover = myth.
    3.0,   # August   - peak abandonment. Martolda is alone.
    0.4,   # September- everyone back. false sense of security.
    0.7,   # October  - Q3 close
    1.3,   # November - pre-December dread sets in
    3.5,   # December - the cruellest month. coat stays on hook permanently.
])

seasonality = np.tile(seasonal_pattern, 3)

# Residuals: the universe's discretionary cruelty
residuals = np.random.normal(0, 0.6, 36)

# Observed series
y = trend + seasonality + residuals
y = np.clip(y, 1, None)  # emergencies cannot be negative. unfortunately.

df = pd.DataFrame({
    'date': dates,
    'emergencies': y,
    'trend': trend,
    'seasonality': seasonality,
    'residuals': residuals
}).set_index('date')

print("Dataset successfully generated. The author is not surprised by any of these numbers.")
print(f"\nTotal emergencies across 36 months : {df['emergencies'].sum():.0f}")
print(f"Average per month                  : {df['emergencies'].mean():.1f}")
print(f"Worst month on record              : {df['emergencies'].idxmax().strftime('%B %Y')} (
print(f"Best month on record               : {df['emergencies'].idxmin().strftime('%B %Y')} (
```

```
Dataset successfully generated. The author is not surprised by any of these numbers.

Total emergencies across 36 months : 257
Average per month                  : 7.1
Worst month on record              : August 2024 (11.2 emergencies)
Best month on record               : February 2022 (4.5 emergencies)
```

In [2]:

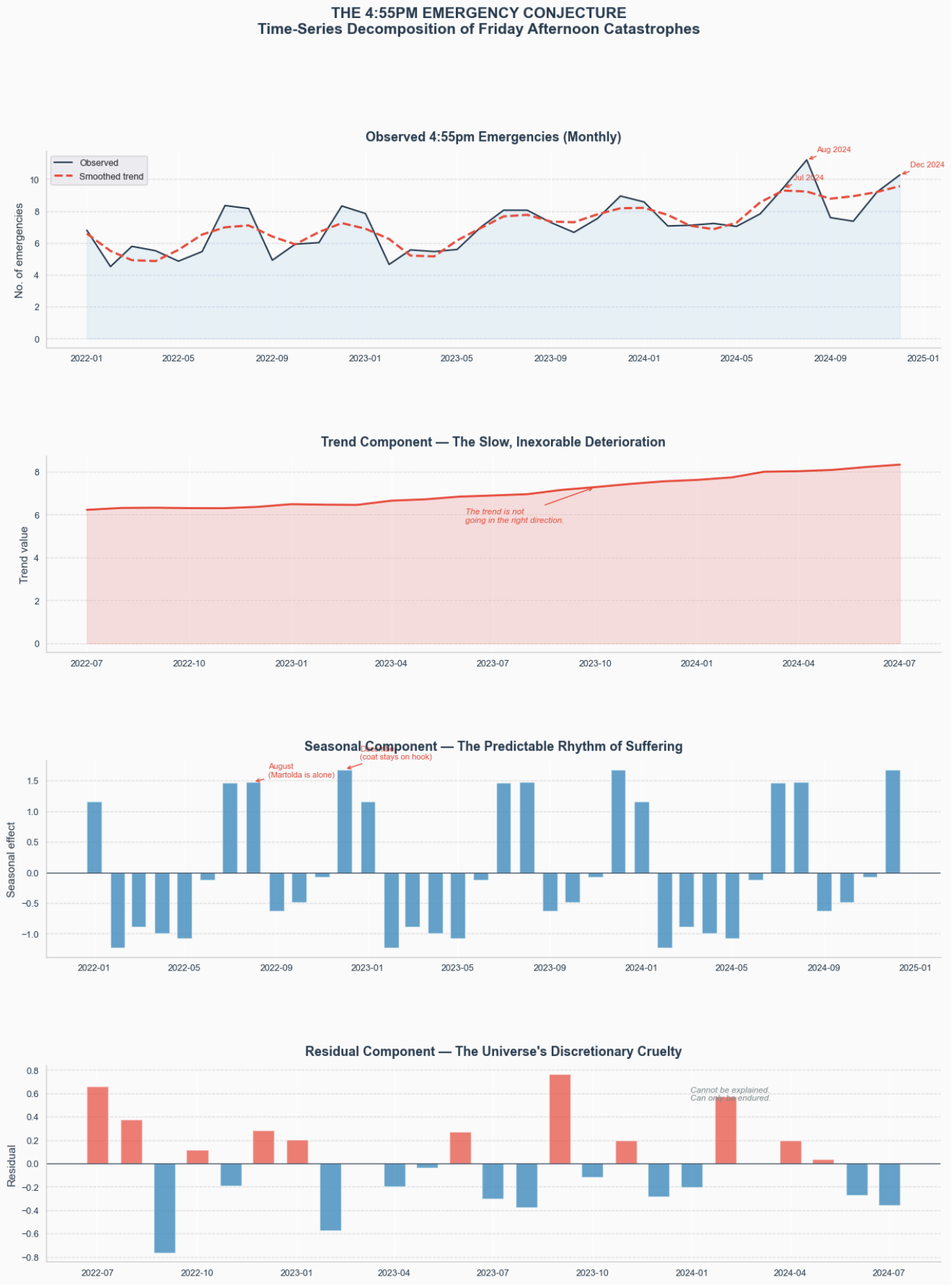
```python
# Manual additive decomposition

# Step 1: Trend via centred 12-month moving average
ma12 = df['emergencies'].rolling(window=12, center=True).mean()

# Step 2: De-trended series
detrended = df['emergencies'] - ma12

# Step 3: Seasonal component - average by month across all years
month_avgs = detrended.groupby(detrended.index.month).mean()
seasonal_extracted = pd.Series(
    [month_avgs[m] for m in df.index.month],
    index=df.index
)

# Step 4: Residuals
resid_extracted = df['emergencies'] - ma12 - seasonal_extracted
resid_std = resid_extracted.dropna().std()

# Step 5: Smoothed trend for visualisation
trend_smooth = savgol_filter(df['emergencies'].values, window_length=7, polyorder=2)

# ── Forecast: 12 months ahead ──────────────────────────────────────
forecast_dates = pd.date_range(start='2025-01-01', periods=12, freq='MS')
trend_slope = (trend[-1] - trend[0]) / (len(trend) - 1)
forecast_trend = trend[-1] + trend_slope * np.arange(1, 13)
forecast_seasonal = seasonal_pattern  # repeat the known seasonal cycle
forecast_mean = forecast_trend + forecast_seasonal

# 95% CI widens with horizon h
ci_widths = resid_std * 1.96 * np.sqrt(np.arange(1, 13))
forecast_upper = forecast_mean + ci_widths
forecast_lower = np.clip(forecast_mean - ci_widths, 0, None)

print("Decomposition complete.")
print(f"Residual std dev (sigma_R)  : {resid_std:.3f}")
print(f"\n12-Month Forecast (2025):")
print("-" * 52)
month_names = ['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
for m, mn, lo, hi in zip(month_names, forecast_mean, forecast_lower, forecast_upper):
    print(f"  {m} 2025:  forecast {mn:5.1f}   CI [{lo:.1f} – {hi:.1f}]")
print("-" * 52)
print(f"  Dec 2025 upper bound: {forecast_upper[-1]:.1f}")
print(f"  (statistically indistinguishable from living at the office)")
```

```
Decomposition complete.
Residual std dev (sigma_R)  : 0.372

12-Month Forecast (2025):
------------------------------------------------------
  Jan 2025:  forecast  10.1   CI [9.4 – 10.8]
  Feb 2025:  forecast   8.2   CI [7.2 – 9.2]
  Mar 2025:  forecast   9.0   CI [7.7 – 10.3]
  Apr 2025:  forecast   8.2   CI [6.7 – 9.7]
  May 2025:  forecast   8.6   CI [7.0 – 10.2]
  Jun 2025:  forecast   9.2   CI [7.4 – 11.0]
  Jul 2025:  forecast  11.0   CI [9.1 – 12.9]
  Aug 2025:  forecast  11.3   CI [9.2 – 13.4]
  Sep 2025:  forecast   8.8   CI [6.6 – 11.0]
  Oct 2025:  forecast   9.2   CI [6.9 – 11.5]
  Nov 2025:  forecast   9.9   CI [7.5 – 12.3]
  Dec 2025:  forecast  12.2   CI [9.7 – 14.7]
------------------------------------------------------
  Dec 2025 upper bound: 14.7
  (statistically indistinguishable from living at the office)
```

In [3]:

```python
# Colour palette: professional but slightly resigned
COLOR_MAIN   = '#2C3E50'
COLOR_TREND  = '#E74C3C'
COLOR_SEASON = '#2980B9'
COLOR_RESID  = '#7F8C8D'
COLOR_FILL   = '#AED6F1'
COLOR_ANNOT  = '#E74C3C'
COLOR_FORE   = '#8E44AD'
COLOR_CI     = '#D7BDE2'

def format_ax(ax, ylabel, title):
    ax.set_facecolor('#FAFAFA')
    ax.set_ylabel(ylabel, fontsize=11, color=COLOR_MAIN)
    ax.set_title(title, fontsize=13, fontweight='bold', color=COLOR_MAIN, pad=10)
    ax.tick_params(colors=COLOR_MAIN, labelsize=9)
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.spines['left'].set_color('#CCCCCC')
    ax.spines['bottom'].set_color('#CCCCCC')
    ax.yaxis.grid(True, linestyle='--', alpha=0.5, color='#CCCCCC')
    ax.set_axisbelow(True)

fig = plt.figure(figsize=(16, 20))
fig.patch.set_facecolor('#FAFAFA')
gs = gridspec.GridSpec(4, 1, hspace=0.55)

# Panel 1: Observed series
ax1 = fig.add_subplot(gs[0])
ax1.fill_between(df.index, df['emergencies'], alpha=0.25, color=COLOR_FILL)
ax1.plot(df.index, df['emergencies'], color=COLOR_MAIN, linewidth=1.5, label='Observed')
ax1.plot(df.index, trend_smooth, color=COLOR_TREND, linewidth=2.2,
         linestyle='--', label='Smoothed trend')
format_ax(ax1, 'No. of emergencies', 'Observed 4:55pm Emergencies (Monthly)')
ax1.legend(fontsize=9, framealpha=0.7)
for idx in df['emergencies'].nlargest(3).index:
    ax1.annotate(idx.strftime('%b %Y'),
                 xy=(idx, df.loc[idx, 'emergencies']),
                 xytext=(10, 8), textcoords='offset points',
                 fontsize=8, color=COLOR_ANNOT,
                 arrowprops=dict(arrowstyle='->', color=COLOR_ANNOT, lw=1))

 # Panel 2: Trend component
```

```
ax2 = fig.add_subplot(gs[1])
valid_trend = ma12.dropna()
ax2.plot(valid_trend.index, valid_trend, color=COLOR_TREND, linewidth=2)
ax2.fill_between(valid_trend.index, valid_trend, alpha=0.15, color=COLOR_TREND)
format_ax(ax2, 'Trend value', 'Trend Component — The Slow, Inexorable Deterioration')
mid_idx = valid_trend.index[len(valid_trend)//2 + 3]
ax2.annotate('The trend is not\ngoing in the right direction.',
             xy=(mid_idx, valid_trend[mid_idx]),
             xytext=(-130, -35), textcoords='offset points',
             fontsize=8.5, color=COLOR_ANNOT, style='italic',
             arrowprops=dict(arrowstyle='->', color=COLOR_ANNOT, lw=1))

# Panel 3: Seasonal component
ax3 = fig.add_subplot(gs[2])
ax3.bar(df.index, seasonal_extracted, color=COLOR_SEASON, alpha=0.7, width=20)
ax3.axhline(0, color=COLOR_MAIN, linewidth=0.8)
format_ax(ax3, 'Seasonal effect', 'Seasonal Component — The Predictable Rhythm of Suffering')
first_dec = seasonal_extracted[seasonal_extracted.index.month == 12].index[0]
first_aug = seasonal_extracted[seasonal_extracted.index.month == 8].index[0]
ax3.annotate('December\n(coat stays on hook)',
             xy=(first_dec, seasonal_extracted[first_dec]),
             xytext=(15, 10), textcoords='offset points',
             fontsize=8, color=COLOR_ANNOT,
             arrowprops=dict(arrowstyle='->', color=COLOR_ANNOT, lw=1))
ax3.annotate('August\n(Martolda is alone)',
             xy=(first_aug, seasonal_extracted[first_aug]),
             xytext=(15, 5), textcoords='offset points',
             fontsize=8, color=COLOR_ANNOT,
             arrowprops=dict(arrowstyle='->', color=COLOR_ANNOT, lw=1))

# Panel 4: Residuals
ax4 = fig.add_subplot(gs[3])
valid_resid = resid_extracted.dropna()
colors_r = [COLOR_ANNOT if v > 0 else COLOR_SEASON for v in valid_resid]
ax4.bar(valid_resid.index, valid_resid, color=colors_r, alpha=0.7, width=20)
ax4.axhline(0, color=COLOR_MAIN, linewidth=0.8)
format_ax(ax4, 'Residual', "Residual Component — The Universe's Discretionary Cruelty")
ax4.annotate('Cannot be explained.\nCan only be endured.',
             xy=(0.72, 0.82), xycoords='axes fraction',
             fontsize=8.5, color=COLOR_RESID, style='italic')

fig.suptitle(
    'THE 4:55PM EMERGENCY CONJECTURE\nTime-Series Decomposition of Friday Afternoon Catastropl
    fontsize=15, fontweight='bold', color=COLOR_MAIN, y=0.98
)

plt.savefig('friday_emergency_decomposition.png', dpi=150,
            bbox_inches='tight', facecolor='#FAFAFA')
plt.show()
print("Decomposition chart saved.")
```

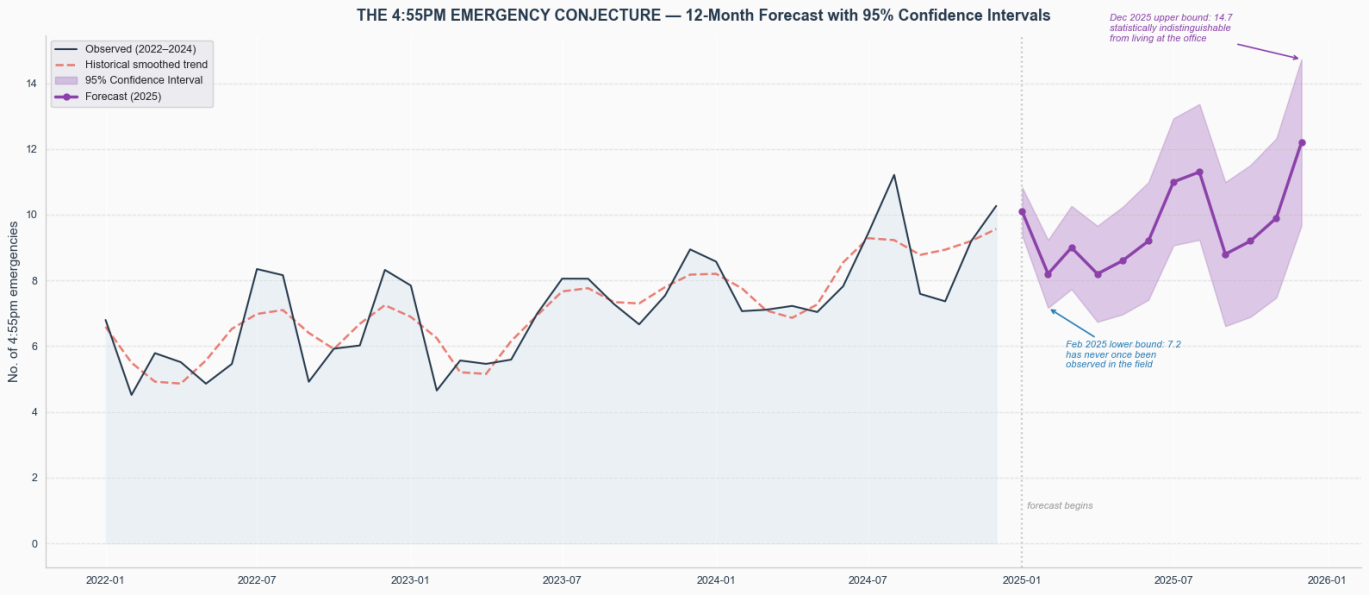


Decomposition chart saved.

In [4]:

```
fig2, ax = plt.subplots(figsize=(16, 7))
fig2.patch.set_facecolor('#FAFAFA')
ax.set_facecolor('#FAFAFA')

# Historical observed series
ax.fill_between(df.index, df['emergencies'], alpha=0.15, color=COLOR_FILL)
ax.plot(df.index, df['emergencies'], color=COLOR_MAIN, linewidth=1.5,
```

```
        label='Observed (2022–2024)', zorder=3)
ax.plot(df.index, trend_smooth, color=COLOR_TREND, linewidth=1.8,
        linestyle='--', alpha=0.7, label='Historical smoothed trend')

# Vertical divider: history vs forecast
ax.axvline(pd.Timestamp('2025-01-01'), color='#CCCCCC', linewidth=1.5,
           linestyle=':', zorder=2)
ax.text(pd.Timestamp('2025-01-01'), ax.get_ylim()[0] if ax.get_ylim()[0] > 0 else 1,
        '  forecast begins', fontsize=8, color='#999999', style='italic', va='bottom')

# Forecast: CI shading, then mean line
ax.fill_between(forecast_dates, forecast_lower, forecast_upper,
                alpha=0.25, color=COLOR_FORE, label='95% Confidence Interval')
ax.plot(forecast_dates, forecast_mean, color=COLOR_FORE, linewidth=2.5,
        marker='o', markersize=5, label='Forecast (2025)', zorder=4)

# Annotate December 2025 upper bound
dec_date = forecast_dates[-1]
ax.annotate(
    f'Dec 2025 upper bound: {forecast_upper[-1]:.1f}\nstatistically indistinguishable\nfrom l
    xy=(dec_date, forecast_upper[-1]),
    xytext=(-160, 15), textcoords='offset points',
    fontsize=8, color=COLOR_FORE, style='italic',
    arrowprops=dict(arrowstyle='->', color=COLOR_FORE, lw=1.2)
)

# Annotate February 2025 lower bound
feb_date = forecast_dates[1]
ax.annotate(
    f'Feb 2025 lower bound: {forecast_lower[1]:.1f}\nhas never once been\nobserved in the fie
    xy=(feb_date, forecast_lower[1]),
    xytext=(15, -50), textcoords='offset points',
    fontsize=8, color=COLOR_SEASON, style='italic',
    arrowprops=dict(arrowstyle='->', color=COLOR_SEASON, lw=1.2)
)

# Formatting
ax.set_title(
    'THE 4:55PM EMERGENCY CONJECTURE — 12-Month Forecast with 95% Confidence Intervals',
    fontsize=13, fontweight='bold', color=COLOR_MAIN, pad=12
)
ax.set_ylabel('No. of 4:55pm emergencies', fontsize=11, color=COLOR_MAIN)
ax.tick_params(colors=COLOR_MAIN, labelsize=9)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_color('#CCCCCC')
ax.spines['bottom'].set_color('#CCCCCC')
ax.yaxis.grid(True, linestyle='--', alpha=0.4, color='#CCCCCC')
ax.set_axisbelow(True)
ax.legend(fontsize=9, framealpha=0.8, loc='upper left')

plt.tight_layout()
plt.savefig('friday_emergency_forecast.png', dpi=150,
            bbox_inches='tight', facecolor='#FAFAFA')
plt.show()
print("Forecast chart saved.")
print("The author has reviewed the December 2025 upper bound. She is considering a career cha
```




```
Forecast chart saved.
The author has reviewed the December 2025 upper bound. She is considering a career change.
```

In [5]:

```python
# Summary statistics
print("=" * 62)
print("   4:55PM EMERGENCY CONJECTURE — SUMMARY STATISTICS")
print("=" * 62)

monthly_avg = df.groupby(df.index.month)['emergencies'].mean()
month_names_short = ['Jan','Feb','Mar','Apr','May','Jun',
                     'Jul','Aug','Sep','Oct','Nov','Dec']

print("\nHistorical average emergencies by month (2022-2024):")
print("-" * 44)
for m, avg in zip(month_names_short, monthly_avg):
    bar = '\u2588' * int(avg)
    flag = " <- peak suffering" if avg == monthly_avg.max() else ""
    flag = " <- brief mercy"   if avg == monthly_avg.min() else flag
    print(f"  {m}: {avg:5.1f}  {bar}{flag}")

print("\n2025 Forecast:")
print("-" * 44)
for m, mn, lo, hi in zip(month_names_short, forecast_mean, forecast_lower, forecast_upper):
    print(f"  {m} 2025: {mn:5.1f}  CI [{lo:.1f} – {hi:.1f}]")

print("\nOverall statistics:")
print("-" * 44)
print(f"  Mean monthly (historical)  : {df['emergencies'].mean():.2f}")
print(f"  Mean monthly (forecast)    : {forecast_mean.mean():.2f}")
print(f"  Residual std dev (sigma_R) : {resid_std:.3f}")
print(f"  Trend direction            : Upward. Significantly.")
print(f"  Author's response          : resignation (ongoing)")
print("=" * 62)
```

```
============================================================
   4:55PM EMERGENCY CONJECTURE — SUMMARY STATISTICS
============================================================

Historical average emergencies by month (2022-2024):
-------------------------------------------
  Jan:   7.7  ███████
  Feb:   5.4  ████▌ <- brief mercy
  Mar:   6.2  ██████
  Apr:   6.1  ██████
  May:   5.8  ██████
  Jun:   6.8  ██████
  Jul:   8.6  ████████▌
  Aug:   9.1  █████████
  Sep:   6.6  ██████
  Oct:   6.7  ██████
  Nov:   7.6  ███████
  Dec:   9.2  ████████▌ <- peak suffering

2025 Forecast:
-------------------------------------------
  Jan 2025:  10.1  CI [9.4 – 10.8]
  Feb 2025:   8.2  CI [7.2 – 9.2]
  Mar 2025:   9.0  CI [7.7 – 10.3]
  Apr 2025:   8.2  CI [6.7 – 9.7]
  May 2025:   8.6  CI [7.0 – 10.2]
  Jun 2025:   9.2  CI [7.4 – 11.0]
  Jul 2025:  11.0  CI [9.1 – 12.9]
  Aug 2025:  11.3  CI [9.2 – 13.4]
  Sep 2025:   8.8  CI [6.6 – 11.0]
  Oct 2025:   9.2  CI [6.9 – 11.5]
  Nov 2025:   9.9  CI [7.5 – 12.3]
  Dec 2025:  12.2  CI [9.7 – 14.7]

Overall statistics:
-------------------------------------------
  Mean monthly (historical)  : 7.14
  Mean monthly (forecast)    : 9.64
  Residual std dev (sigma_R) : 0.372
  Trend direction            : Upward. Significantly.
  Author's response          : resignation (ongoing)
============================================================
```