Project Methodology: (Path B) GNN with LLM-Generated Inputs & Explainability

This document outlines the end-to-end process for a state-of-the-art stock prediction model. The methodology is based on a **Spatio-Temporal Graph Neural Network (GNN)**.

- **Core Idea:** To model the 30 stocks as a dynamic network. The model learns how quantitative stock features (the "nodes") interact and "flow" along a "relationship matrix" (the "edges") that is generated daily from financial news.
- **Academic Precedent:** This approach is directly inspired by SOTA papers like MDGNN (Multi-Relational Dynamic Graph), ECHO-GL (Earnings Calls-Driven Graph), and GAT-AGNN (Graph Attention Network).

The workflow is broken into six key stages.

Stage 1: Preprocessing (The Three Input Tensors)

This is the most critical and time-consuming stage. You must create three separate, massive datasets (tensors) that are all perfectly aligned by date. Assuming 5 years of data (e.g., 1795 usable samples after a 30-day lookback) and 30 stocks:

1. Input X: The Node Feature Matrix (Quantitative Data)

- **What it is:** The quantitative features for *each* of your 30 stocks for the *previous 30 days*.
- **Purpose:** Provides the "state" of each node in the graph.
- **Shape:** (1795, 30, 8)
  - **1795 (Samples):** The number of trading days.
  - **30 (Nodes):** The 30 stocks in your universe (e.g., FBM KLCI).
  - **8 (Features):** The set of features for each stock, e.g.:
    1. Daily Return
    2. Normalized Trading Volume
    3. RSI (14-day)
    4. MACD
    5. Volatility (20-day)
    6. FBM KLCI Daily Return (Benchmark)
    7. MYR/USD Daily Return (Macro)
    8. BNM OPR Rate (Macro)
    9. Sentiment Scores

2. Input A: The Dynamic Relationship Matrix (LLM-Powered News)

- **What it is:** Your core idea. A 30x30 matrix, **different for every single day**, that defines the *relationship* between all 30 companies based on that day's news.
- **Purpose:** Provides the *graph structure* (the "edges") for the GNN.

- **Shape:** (1795, 30, 30)
  - **1795 (Samples):** A new matrix for each day.
  - **30 (Nodes):** Company A.
  - **30 (Nodes):** Company B.

Daily LLM Pipeline to Build This Input (Run 1,795 times):
For each day t in your dataset:

1. **Fetch News:** Use your 30-year historical API to get all articles for that day.
2. **Filter for Co-Mentions:** Run NER (e.g., spaCy) on all articles. Keep *only* the articles that mention **two or more** of your 30 stocks. (This solves the "Market Report" noise flaw).
3. **Extract Relation (The LLM Step):** For each of these high-signal articles:
   - **Identify Pair:** e.g., article mentions MAYBANK.KL and CIMB.KL.
   - **Call generateContent:** Programmatically call the gemini-2.5-flash-preview-09-2025 API.
   - **System Prompt:**
     You are a financial analyst. Your task is to read a news article and find the single, most important semantic relationship between two specified companies. Respond ONLY with a JSON object with two keys:
     1. "relationship_type": (e.g., 'Partnership', 'Competition', 'Lawsuit', 'Supplier', 'NoRelationship')
     2. "sentiment_score": A float from -1.0 to +1.0 describing the sentiment OF THAT RELATIONSHIP.

   - **User Prompt:**
     Article: "Maybank reports record profits but announces new lawsuit against CIMB for fraud."
     Company A: "Maybank"
     Company B: "CIMB"

   - **generationConfig (Crucial):**

```
{
  "responseMimeType": "application/json",
  "responseSchema": {
    "type": "OBJECT",
    "properties": {
      "relationship_type": { "type": "STRING" },
      "sentiment_score": { "type": "NUMBER" }
    }
  }
}
```

   - **LLM's JSON Output:** {"relationship_type": "Lawsuit", "sentiment_score": -0.8}
4. **Build Daily Matrix:** Create a blank 30x30 matrix. When you get the LLM output, you fill

the matrix:
    - matrix[MAYBANK, CIMB] = -0.8
    - matrix[CIMB, MAYBANK] = -0.8 (assuming symmetry)
5. **Aggregate:** If multiple articles mention the same pair, average their (now *correct*) relationship scores. This final matrix is your A_t for this day.

Stage 2: Model Architecture (The Graph *Attention* Network)

This is the critical change. To add the "explanation function" inspired by the PEN paper, you will not use a standard GNN. You will use a **Graph Attention Network (GAT)**, as seen in the GAT-AGNN paper.

- **Standard GNN (GCN):** This is a "black box" engine. It blindly trusts and multiplies your input A (the relationship matrix). It provides **no explanation**.
- **Graph Attention Network (GAT):** This is an "explainable" engine. It takes your input A as "advice" but then *learns* its own internal **"importance scores" (attention weights, alpha)** for every relationship, based on the context from Input X.

The alpha matrix *is* your explanation.

Your model architecture will be a **Spatio-Temporal GAT**:

1. **Input:** The model takes a "slice" of your data for day t:
    - The (30, 8) Node Features (X_t).
    - The (30, 30) Relationship Matrix (A_t).
2. **GAT Layer:** The GAT layer (GAT(X_t, A_t)) dynamically calculates its internal alpha weights and mixes the node features based on those weights. The output is a new (30, 8) tensor where each stock's features now also contain *attention-weighted* information from its "neighbors."
3. **Temporal Layer (LSTM):** The output of the GAT is fed into an LSTM or GRU layer. This layer's job is to learn patterns from the *sequence* of these daily, graph-enhanced features.

Stage 3: Prediction Output (Regression)

The model must be trained to predict the future price change.

- **Final Layer:** The output of the temporal layer is fed to a Dense layer with **30 neurons** and a **linear activation**.
- **The Output (Label y):** A 30x1 vector of continuous numbers.
    - **Example:** [+0.012, -0.005, +0.001, ...]
- **What it Represents:** The predicted **5-day (or n-day) future return** for each of the 30 stocks.
- **Loss Function:** You will train the entire GAT model by minimizing the **Mean Squared Error (MSE)** between your predicted vector and the true future returns.

Stage 4: Portfolio Allocation (The Markowitz Optimizer)

This is the final step of your *daily* simulation. Your GNN has given you the "profit" signal; now you use it.

- **Goal:** To find the optimal portfolio (set of weights) that gives the highest **Sharpe Ratio** (best risk-adjusted return).
- **Action:** You will use the **PyPortfolioOpt** Python library.
- **The Optimizer's Inputs:**
  1. **Expected Returns (μ):** The 30x1 output vector *from your GNN* (Stage 3).
  2. **Risk Matrix (Σ):** The 30x30 historical covariance matrix you built *in Stage 1*.
  3. **Risk-Free Rate (R_f):** The 10-Year Malaysian MGS yield.
- **The Optimizer's Output:** A 30x1 vector of **optimal weights** (e.g., [0.10, 0.05, 0.0, 0.08, ...]). This is your **final "Earning Portfolio Allocation"** for that day.

Stage 5: Backtesting & Validation

- **Process:** You must use **Walk-Forward Validation**.
  - **Example:** Train your GAT on 2018-2022 data.
  - Run your full (Stage 4 + Stage 5) pipeline for every day in 2023, saving your portfolio weights.
  - *Retrain* your GAT on 2018-2023 data.
  - Run your full pipeline for every day in 2024.
- **Evaluation:** Calculate the final Sharpe Ratio, Cumulative Return, and Max Drawdown of your strategy. Compare this directly to the FBM KLCI (your benchmark).

Stage 6: Explainability (The "PEN" Method)

This is your final research step, made possible by using a GAT in Stage 2.

- **Action:** After your model makes a prediction (e.g., "Buy Maybank"), you can programmatically **extract the internal alpha (attention) weights** from the GAT layer for the "Maybank" node.
- **The Explanation:** These weights are your "Vector of Salience" for relationships. You can now create a table in your FYP paper that *proves* why your model acted.

Example Explanation:

> "The model's decision to buy Maybank was primarily driven by its learned attention. As shown, it placed **80% importance** on the 'Lawsuit' relationship with CIMB and **15% importance** on the 'Partnership' relationship with YTL. It correctly learned to ignore the 'Market Report' noise from Tenaga, assigning it only 5% importance."

Relationship (Edge)	**LLM-Input Score A**	**GAT-Learned Importance alpha**
Maybank $\leftrightarrow$	-0.8 (Lawsuit)	**0.80 (80% Importance)**

CIMB		
Maybank $\leftrightarrow$ YTL	+0.7 (Partnership)	**0.15 (15% Importance)**
Maybank $\leftrightarrow$ Tenaga	+0.6 (Market Report)	**0.05 (5% Importance)**