StorePulse

Demand Forecasting Automation Platform using NB-INGARCH Models

MTech Technical Presentation - Complete System Architecture

Forecast Generation Flow

How predictions are generated when user requests a forecast.

sequenceDiagram autonumber participant U as User participant UI as Frontend participant API as ForecastRoute participant Cache as Cache Layer participant FS as Forecast Service participant DB as Database participant Model as INGARCH Model U->>UI: Navigate to Forecast Page UI->>API: GET /forecast?days=7&mode=lite rect rgb(240, 248, 255) note right of API: Cache Check API->>Cache: Check for cached forecast alt Cache Hit Cache-->>API: Return cached data API-->>UI: JSON response (instant) else Cache Miss API->>FS: Generate fresh forecast end end rect rgb(255, 243, 224) note right of FS: Load Model FS->>DB: Query latest model metadata DB-->>FS: Model info FS->>FS: Load ingarch_model.joblib end rect rgb(232, 245, 233) note right of FS: Feature Building FS->>DB: Fetch last 365 days of visits DB-->>FS: Historical data FS->>FS: For each forecast day (1-7): FS->>FS: Compute lag features FS->>FS: Check holiday calendar FS->>FS: Calculate rolling stats FS->>FS: Feature matrix ready end rect rgb(252, 228, 236) note right of FS: Prediction FS->>Model: predict(X_features) Model->>Model: Apply autoregressive terms Model->>Model: Apply exogenous effects Model->>Model: Sample from NB distribution Model-->>FS: Raw predictions FS->>FS: Calculate bounds (P10, P50, P90) FS-->>API: 7-day forecast with uncertainty end rect rgb(227, 242, 253) note right of API: Post-Processing API->>API: Calculate staffing needs API->>API: Generate inventory alerts API->>Cache: Save forecast (TTL: 1 hour) API-->>UI: Complete JSON response UI-->>U: Display charts end

StorePulse - Demand Forecasting Automation Platform

MTech Project | NB-INGARCH Implementation | Verified Metrics

All diagrams, metrics, and architecture details are based on actual implementation