## **Tech Stack Flow**

#### 1. Data Ingestion Layer (External Signals)

- News & Event APIs (Google News API, GDELT, Bloomberg, or sample mock feeds) → global disruptions.
- Weather APIs (Google Weather API, OpenWeatherMap) → floods, storms, droughts.
- Economic/Trade Data (World Bank API, UN Trade Data, commodity price feeds).
- Optional: IoT/Warehouse mock sensor data (temperature, stock levels).

#### **Data Sources**









# News & Event APIs

APIs like Google News and Bloomberg provide data on global disruptions.

#### **Weather APIs**

APIs like Google
Weather and
OpenWeatherMap
provide data on
floods, storms, and
droughts.

## **Economic/Trade Data**

APIs like World Bank and UN Trade Data provide economic and trade information.

#### IoT/Warehouse Data

Optional data from IoT sensors provides temperature and stock level information.

Made with > Napkin

#### 2. Preprocessing & Event Structuring

- Cloud Function (Firebase Cloud Functions): fetches raw feeds.
- Gemini API:
  - Summarizes long news reports → "Port strike in LA, impacting exports."
  - Classifies **relevance** (High, Medium, Low impact).
  - Extracts entities (location, commodity, supplier).

#### **Gemini API Functions**



#### **Summarization**

Summarizes long news reports into concise summaries.



# **Relevance Classification**

Classifies the relevance of news reports as high, medium, or low impact.



#### **Entity Extraction**

Extracts key entities such as location, commodity, and supplier from news reports.

Made with ≽ Napkin

#### 3. Risk Analysis Engine

- **Gemini API** (reasoning layer):
  - Predicts disruption severity (short delay vs. global shortage).
  - Cross-checks against historical patterns (toilet paper shortage, chip shortage).
  - Suggests contingency plans: "Switch to Supplier B in Vietnam."

## **Predicting Disruption Severity**



#### Historical Crosscheck

Cross-check against historical patterns

# **Contingency Suggestion**

Suggest relevant contingency plans

# **Predicted Disruption**

Known severity of disruption



#### 4. Data Storage & Sync (Firebase)

- Firestore: stores disruption events, risk scores, supplier database.
- **Firebase Authentication**: users (logistics managers, companies) login to see custom dashboards.
- Firebase Cloud Messaging: push real-time alerts  $\rightarrow$  " $\triangle$ !! Risk Alert: Typhoon in Taiwan affecting semiconductors."

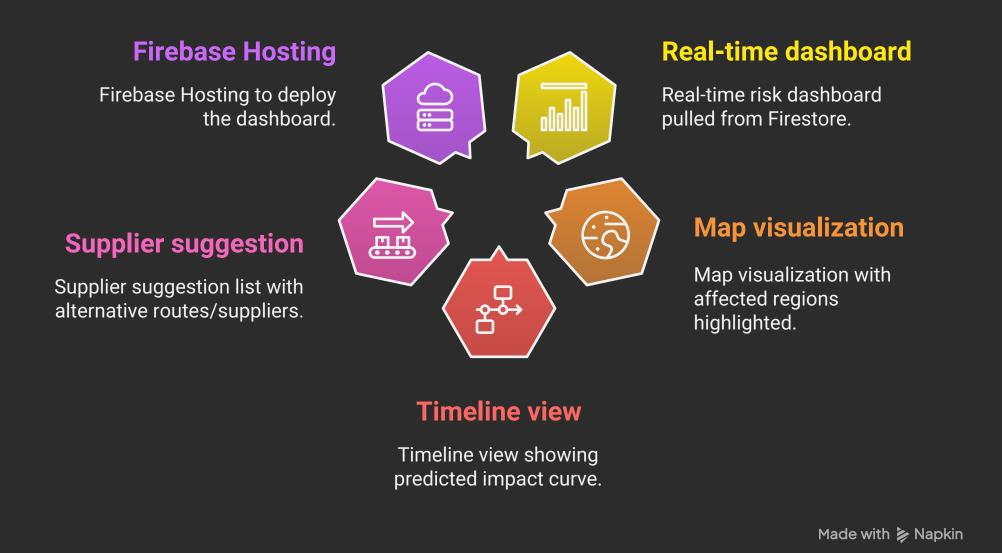
#### **Firebase Services**



#### 5. Frontend / Dashboard

- **React / Flutter Web** (for cross-platform UI):
  - Real-time risk dashboard (pulled from Firestore).
  - Map visualization → affected regions highlighted.
  - Timeline view → predicted impact curve.
  - Supplier suggestion list → alternative routes/suppliers.
- Firebase Hosting → deploy the dashboard.

#### **Dashboard features**



#### 6. Al Interaction Layer

- Chat Interface (Gemini API + Firebase):
  - User asks: "What's the biggest risk to electronics supply in the next month?"
  - Gemini responds with structured insights + supporting news sources.

# Semini-Firebase Interaction User Input The user's question initiates the process. Facilitates data storage and synchronization. Gemini API Provides structured insights and news sources. Made with ≱ Napkin

Flow Diagram (Simplified)

External Data Sources (News, Weather, Trade APIs) → Firebase Cloud Functions (fetch & clean) → Gemini API (summarize, classify, predict, plan) → Firestore Database (store disruptions, suppliers, risk scores) → Dashboard (React/Flutter) + Push Alerts (Cloud Messaging)

### **Tech Stack for Risk Management**



#### **Gemini API**

Summarizes and classifies data for insights

Made with > Napkin

#### Suggested Stack

- Frontend: React.js (or Flutter Web for cross-platform).
- Backend: Firebase (Firestore, Cloud Functions, Hosting, Auth, Cloud Messaging).
- AI: Gemini API (reasoning + text understanding).
- Data APIs: Google News, Weather, Trade Data.
- Visualization: Mapbox or Google Maps API (to show affected regions).

## **Project technologies**

