Explanation of Approach

- **REST Controller (AlertController):** This class handles the incoming HTTP request. It uses @PathVariable to extract the companyId from the URL and delegates the business logic to the AlertService. This keeps the controller lightweight and focused on handling web requests.
- **Business Logic Service (AlertService):** This is where the core logic resides. It performs a series of steps to filter the data according to the business rules:
  - It first queries the database to get a list of all productIds that have had recent sales (within the last 30 days) for the specified company. This addresses the "only alert for products with recent sales" rule.
  - It then fetches all relevant inventory items and thresholds.
  - It streams through the relevant inventory data, filtering for items where the quantity is below the productType specific threshold.
  - For each low-stock item, it performs additional lookups to get the associated product and supplier information, and calculates the daysUntilStockout based on a simple assumed sales rate.
  - Finally, it maps the data into a list of AlertDto objects, which are then returned to the controller.
- **Data Access Repositories:** We use Spring Data JPA repositories to simplify database interactions. The OrderItemRepository contains a custom @Query method to efficiently find products with recent sales activity directly from the database, which is more performant than pulling all data into memory.
- **Data Transfer Objects (DTOs):** The AlertDto and AlertsResponse classes are used to format the data exactly as required by the API specification. This ensures a clean contract between the backend and the frontend, hiding the underlying database structure.
- **Edge Case Handling:** The code includes checks for empty results (if (productIdsWithRecentSales.isEmpty())) to prevent unnecessary database operations and returns a clean, empty list when there are no alerts. It also handles the case where a product might not have an associated supplier, returning null for the supplier object instead of an error.