Key Components

1. **C# Web API** for exposing endpoints for the store application.
2. **Azure Functions** for handling asynchronous operations and event-driven tasks.
3. **Azure SQL Database** or **Cosmos DB** (based on scalability requirements) for the Cloud Database.
4. **Azure Blob Storage** for managing media files like product images.
5. **Event Grid** for orchestrating events between components (e.g., product creation, updates).
6. **Azure Key Vault** for securely storing sensitive information like database connection strings.

Architecture Overview

The architecture is centered around a Web API and Azure Functions that interact with an Azure-based database and other services. Here's a high-level overview of each component:

- **Web API**: Provides REST endpoints for managing products, categories, and orders.
- **Azure Functions**: Operates in response to events, handling background tasks such as sending notifications, processing orders, and updating product data.
- **Database Layer**: Hosted on Azure SQL Database or Cosmos DB, depending on requirements for global distribution and high availability.
- **Blob Storage**: Used for storing media files associated with products.
- **Event Grid**: Facilitates communication between the Web API and Azure Functions.

Detailed Design

1. C# Web API (Core Endpoints)

The Web API will provide a RESTful interface for basic store operations. Some essential endpoints are:

- GET /api/products: Retrieve a list of products.
- POST /api/products: Create a new product entry.
- PUT /api/products/{id}: Update an existing product.
- DELETE /api/products/{id}: Delete a product entry.
- POST /api/invoice: Create a new order.
- GET /api/Costumers/{id}: Retrieve a Costumer

These endpoints would interact with the database layer and trigger events when actions are performed. The database design could follow a simple relational model or a NoSQL structure:

- **Products Table**: Contains details about each product.

- **Categories Table**: Defines product categories.
- **Orders Table**: Stores information about customer orders.

2. Azure Functions

Azure Functions are ideal for performing background tasks asynchronously. Here are two sample Azure Functions we can use:

- **Function 1: Process/Store Order**
    - Triggered by an Event Grid message when a new order is created.
    - Validates and processes the order, updating the Orders table in the database.
    - Can be configured to send notifications or emails about order confirmation.
- **Function 2: Process Costumer Data**
    - Triggered by a HTTP Trigger
    - Makes a list of the necessary Costumer Data and returns its

3. Database Architecture

To achieve scalability and manageability in the cloud, we recommend using a Tabel Storage instead of an SQL server as the cost for running an Azure SQL Database are enormous comparing to Table Storage. This proof of concept is made in SQL.

4. Blob Storage for Media Assets

For storing images, videos, and other media files related to products, **Azure Blob Storage** is a good fit. Product data in the database would store the URI path to the blob, allowing for easy retrieval when accessing media.

5. Event Grid Integration

Azure Event Grid would serve as a decoupling mechanism, handling notifications between the Web API and Azure Functions:

- When a product is created or updated, an event is published to Event Grid.
- This triggers the Update Inventory function or other related functions.
- An order placed event can similarly trigger the Process Order function, decoupling the web app from the processing tasks.