1. Objective

ABC Technologies has collaborated with HashedIn University to build a platform to manage their data warehouse. They are looking for a centralized management system that can be accessed through APIs. This document describes the business context of the application that needs to be developed for them.

1.1 Business Context

Mr. Hank DEF, President and CEO of ABC Technologies thought of maintaining a centralized management system for data warehouses in ABC Technologies. As a result, ABC Technologies started the project DManage. They have developed Frontend for DManage. As it is affecting their business, they contacted HashedIn University to come up with a platform that helps their Frontend through APIs.

2. Project Plan

Mr. DEF has a critical presentation on **Saturday Evening at 5.00 pm**. We have until Friday evening to complete all the deliverables. The delivery is expected to be high quality, iterative and showing progressive enhancements without waiting for a last-minute integration. This will reduce the risk and keep Mr. DEF informed about the progress schedule instead of a last-minute surprise. There are daily checkpoints twice a day to keep up the progress.

HashedIn By Deloitte technical team has estimated the project to be doable by an experienced engineer in approximately 24 hrs. Given that the HashedIn University team is newly introduced to .NET Core Framework, they are expected to take slightly longer. It's expected that they will heavily prioritize doing the important stuff to make the presentation on Friday successful.

The milestones below show the must-have features and how they should be planned for delivery. The remaining time should be used to complete the functionality and prioritized based on the business context.

3. Functional Requirements

Background:

ABC is a big box company. A big-box retailer is **a retail store that occupies an enormous amount of physical space** and offers a variety of products to its customers. These stores achieve economies of scale by focusing on large sales volumes. These stores often provide good discounts to their customers.

Every data warehouse has below operations:

1. ***Inbound*** - Large Items Purchase happens or seller ship large items and stores by the Data warehouse
2. ***Outbound -*** Customers purchase items in bulk, shipment happens from warehouse to customer store
3. ***Warehouse (Inventory)-*** Manage items in the warehouse, manage reports etc
4. ***System Management -*** Manage Users, Manage master configuration for the warehouse.

Below are the required services in the system, each service has required operations that need to be implemented.

Assumption:

1. Pallet is the group of a product type with predefined quantity.
2. Each Pallet has specific location in the data warehouse
3. Each warehouse has multiple rooms and storage area.
4. product can only be assigned to pallets, each pallet can be placed in the data warehouse specific place.

- **InBound Operation Service:**

  Receive - When an order is received, there are 2 operations required at the Main gate.
  1. Create a dummy order
  2. Verify Order Id
  3. Accept/Reject the order

  Cross Dock: After order approved, arrange items and move into the warehouse. It can have the following operations supported.

  1. Modify the Pallets quantities
  2. Assign the priority to the item.
  3. Assign items to pallets.

  Putaway: Move items to warehouse

  1. Move Items/pallets within warehouse
  2. Api to find out Quantities of items are available in the warehouse.

- **OutBound Operation Service:**

  Generals:

  1. Receive orders from customers.
  2. Override Items priority.
  3. Provide discounts and offers for the customer

  Order Planning:

  1. See items ready for the shipment
  2. Assign trucks for the shipment
  3. Assign driver for the shipment.

  Shipping:

  1. Schedule time and day for the shipment
  2. Manage status of the shipment

Order:

1. Approve/Reject the order
2. Change order quantities.
3. Manage discounts on the orders.

- **Warehouse Management Service**:

  Inventory Management:

  1. View Pallets Details:
  2. View Product Details:
  3. View Product location
  4. Update Product Information
  5. View Orders status
  6. Update/Manage LPN
  7. Search APIs for the Products.
  8. View all the orders and their status.

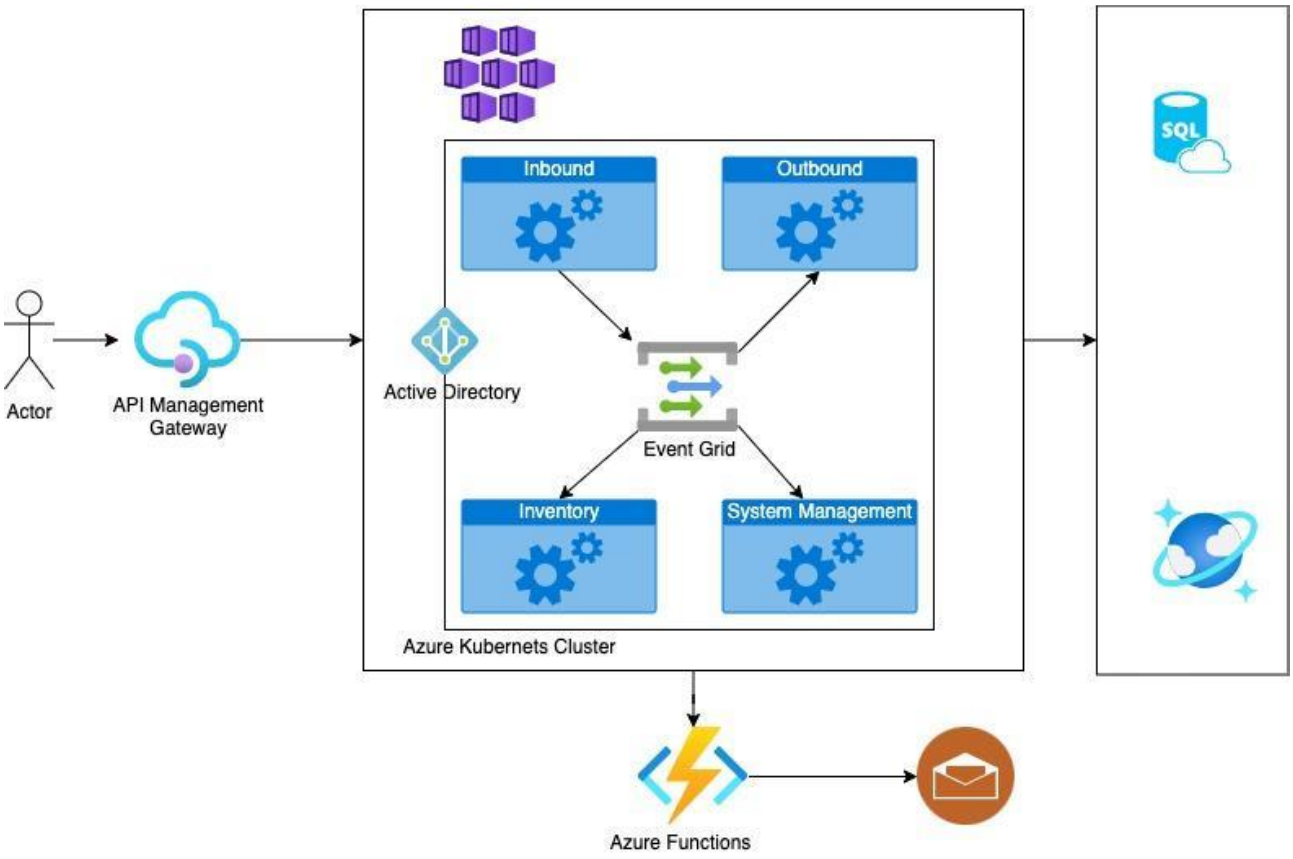
- **System Management Service**:

  1. Define Products type for the warehouse
  2. Define Pallets items quantity
  3. Define Pallet with item type.
  4. Defines Number of Nodes within warehouse
  5. Define LPN for Pallets
  6. https://www.slideshare.net/MattSaragusa/warehouse-storage-managment-system-database-schema (Database schema reference for Inventory)

4. Proposed architecture

There is also a proposed architecture by the ABC which can be taken as reference for the development and deployment.

Note: In this architecture you can use of **Azure service bus**/ RabittMQ/EventGrid



5. Milestones

API Management should have all the implemented APIs in Azure.

Milestone - 1	Clone the project from {boilerplate code link}, Setup the boilerplate project and create a database, required tables for System Management, Add all the master data for Product, Pallets, Nodes, LPN into the system by implementing the System Management APIs.
Milestone - 2	Implement all the APIs in the Inbound Service and APIs.
Milestone - 3	Implement All the APIs for Outbound Service and APIs
Milestone - 4	Implement all the APIs for Inventory Service and APIs
Milestone - 5	Implement Azure AD authentication and 2 Roles Manager, Admin, All Service except System Management can be access by store manager, Admin only should have access of System Management APIs
Milestone - 6	Implement Azure Event grid for Microservice inter communication.
Milestone - 7	Implement search, pagination, caching at required places.
Milestone - 8	Implement Microservice and deploy over AKS (Azure Kubernetes service)
Milestone - 9	Implement proper logging into the application to trace the issue using correlation ID
Milestone - 10	Implement proper Unit test cases for the Solutions.

QUALITY FOCUS
Even though the client doesn't mention quality, the quality focus must be there. Whatever we deliver must be delivered with excellent quality. This is important for getting continued business from PYM Industries and establishing the value HashedIn by Deloitte can deliver to them i.e. "High Quality delivered Quickly"

5. Non-Functional Requirements

- Best Practices
    - Verify your codebase. There shouldn't be any code violation, check using resharper or sonar qube analysis
    - Unit tests handling business cases and Integration test cases should be written for all functionalities.

7. Links for Reading Material

1. TODO WebaPI with Entity framework
2. ORM Queries using Entity framework via LINQ
3. REST Endpoints
4. Authentication and Authorization
5. Linq Pagination, Sorting, Filtering
6. C# Basics