

Amazoom: Automated Warehouse

Metehan Sahin
Kevin Jiang
Lucy Hua
Nic Ricci





Tech Stack

Warehouse Computer

- **C# Console Application**
 - Allows initiation of multiple warehouses
 - Each warehouse initiation creates a new process that handles incoming orders, restocking, as well as robot + truck movement
- **Warehouse Interface**
 - Allows warehouse manager to easily manage warehouses at the clicks of buttons
 - Interface shows movement of robots, inventory and more



Tech Stack

Webserver Database and Interface:

- **ASP.NET (MVC)**
 - Interactive Web UI development
 - Real-time, customizable
- **SQL**
 - Portable database software
 - Fast query processing
- **Dapper**
 - Stores data in data tables
 - Executes queries and maps results from database to project file and vice versa
 - Convenient functions for INSERT, UPDATE, DELETE etc.

ItemID	ItemWeight	ItemVolume	ItemName	Stock		
1	200	50	Potato	14	Edit Details Delete	Add to Cart
15	1300	5000	Bicycle	15	Edit Details Delete	Add to Cart
14	200	150	Bottle	100	Edit Details Delete	Add to Cart
3	5	1	Plastic Bottle	200	Edit Details Delete	Add to Cart
23	100000	100000	Dragon	50	Edit Details Delete	Add to Cart
3	5	5	Bot	10	Edit Details Delete	Add to Cart
6	1000	5000	Skateboard	150	Edit Details Delete	Add to Cart

[Return to Home](#) [Check Out](#)

© 2021 - My ASP.NET Application

```
0 references
public static void UpdateStock(int itemId, int quantity)
{
    // Updating database stock inventory after an order (Warehouse Computer -> Database)
    var connectionString = ("Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=WCDemoDB;Integrated Security=True;Connect Timeout=30;");
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        var itemId = itemId;
        var newQuantity = quantity;
        connection.Execute("UPDATE Item SET Stock = @Stock WHERE ItemID = @ItemID", new { Stock = newQuantity, ItemID = itemId });
    }
}
```



Overall Design

- **Warehouse**
 - **Warehouse Computer**
 - Handles communication and interaction between objects within the warehouse
 - Storage of information regarding robots (active and inactive), trucks (docked, waiting to be docked), items (list, location, and quantity), order queue (incoming, processing, ready for shipping), shelves (empty, non-empty, full and location) and racks
 - **Grid**
 - Shelves, racks and loading docks, loading dimensions from file
 - **Robot**
 - Pathfinding algorithm, collision avoidance, item loading/restocking
 - **Truck**
 - Docking algorithm
- **Webserver**
 - **Client Interface**
 - Product list, cart/orders, manually update inventory
 - **Database**
 - Warehouse inventory, order information



Demo

Client Order Querying

Updating Warehouse Item Inventory

Robot Pathfinding and Collision Avoidance

Warehouse Trucks

Warehouse Flexibility (Multiple units, Dimensions)

+ Other functionalities