**Verifica Spring Boot**

**Contesto**

Consideriamo una azienda che produce modellini giocattolo di automobili.

**Obiettivi dell’applicazione**L’Azienda vuole gestire il calcolo del fabbisogno degli articoli necessari per produrre un ordine di un determinato Prodotto Finito.

**Implementazione WebAPI**

Obiettivo del compito è realizzare una WebAPI per il calcolo dei fabbisogni di una azienda . Le informazioni vanno gestite tramite una base dati: scegliere quale tipo di database utilizzare.

Per ogni Articolo deve essere memorizzato (**TArticoli**):

1. ArticoloID (numero intero generato dal sistema che si incrementa automaticamente: chiave primaria)
2. Nome  (Testo)
3. Tipologia (Testo) (PF=prodotto finito, SL=semilavorato)
4. Giacenza (numero intero)
5. CostoUnitario (Numero)

Caricare a mano i seguenti Articoli:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ArticoloID** | **Nome** | **Tipologia** | **Giacenza** | **CostoUnitario** |
| 1 | Automobile di Tipo A | PF | 4 | 0 |
| 2 | Automobile di Tipo B | PF | 5 | 0 |
| 3 | Ruote | SL | 100 | 0,05 |
| 4 | Assi | SL | 100 | 0,05 |
| 5 | Telaio Tipo A | SL | 100 | 0,1 |
| 6 | Telaio Tipo B | SL | 100 | 0,2 |
| 7 | Carrozzeria Tipo A | SL | 100 | 0,3 |
| 8 | Carrozzeria Tipo B | SL | 100 | 0,4 |
| 9 | Viti | SL | 100 | 0,03 |

Deve essere possibile memorizzare la distinta base (ad un solo livello) per il montaggio dei prodotti finiti (**TLegami**):

1. LegameID (numero intero generato dal sistema che si incrementa automaticamente: chiave primaria)
2. ArticoloID\_padre (numero intero)
3. ArticoloID\_figlio (numero intero)
4. CoefficienteFabbisogno (numero intero)

Caricare a mano la distinta base per Automobile di Tipo A e Automobile di tipo B:

|  |  |  |  |
| --- | --- | --- | --- |
| LegameID | ArticoloID\_padre | ArticoloID\_figlio | CoefficienteFabbisogno |
| 1 | 1 | 3 | 4 |
| 2 | 1 | 4 | 2 |
| 3 | 1 | 5 | 1 |
| 4 | 1 | 7 | 1 |
| 5 | 1 | 9 | 2 |
| 6 | 2 | 3 | 4 |
| 7 | 2 | 4 | 2 |
| 8 | 2 | 6 | 1 |
| 9 | 2 | 8 | 1 |
| 10 | 2 | 9 | 2 |

Deve essere possibile la memorizzazione di Ordini di prodotti finiti (**TOrdini**):

1. OrdineID  (numero intero generato dal sistema che si incrementa automaticamente: chiave primaria)
2. ArticoloID (numero intero) (sarà possibile inserire solo PF quindi solo ArticoloID=1 oppure ArticoloID=2)
3. QuantitaDaProdurre (numero intero)
4. ScaricoEffettuato (true/false)
5. CostoTotaleSemilavorati  (Numero)

Deve essere possibile la memorizzazione di Fabbisogni di ogni articolo per un determinato OrdineID (**TFabbisogni**):

1. FabbisognoID  (numero intero generato dal sistema che si incrementa automaticamente: chiave primaria)
2. OrdineID (numero intero)
3. ArticoloID (numero intero)
4. QuantitaFabbisogno (numero Intero)

Creare le seguenti chiamate mediante Spring Boot (decidere la tipologia http di chiamata):

1. **Inserimento di un nuovo ordine**. Input: ArticoloID (solo PF quindi solo ArticoloID=1 oppure ArticoloID=2)  e QuantitaDaProdurre (numero intero).  La  WebAPI inserisce il nuovo Ordine in TOrdini. Non inserire il CostoTotaleSemilavorati   perché verrà aggiornato in una chiamata successiva
2. **Calcolo dei fabbisogni per un ordine**.  Input: OrdineID.  La WebAPI aggiunge i vari fabbisogni in TFabbisogni per ogni SL  necessario per produrre il PF. Attenzione che se all’OrdineID sono già associati dei fabbisogni-> vanno prima cancellati altrimenti si rischia di avere fabbisogni duplicati per un singolo ordine
3. **Visualizzazione dei fabbisogni di un certo OrdineID:**  Input: OrdineID. La WebAPI visualizza  i fabbisogni  (TFabbisogni) per ogni SL (calcolati in precedenza tramite la funzionalità numero 2)
4. **Scarico Magazzino**: Input:  OrdineID. La WebAPI scarica i fabbisogni dalla TArticoli (fa calare la giacenza di ogni ArticoloID correlato) e memorizza in TOrdini che è stato effettuato lo scarico di magazzino per quel determinato OrdineID. Attenzione che se TOrdini.ScaricoEffettuato=true è necessario comunicare che lo scarico è già stato effettuato e quindi la procedura si interrompe altrimenti si rischia uno scarico magazzino multiplo.
5. **Calcolo del Costo Totale dei Semilavorati per un Ordine**.  Input OrdineID.  La WebAPI calcola il costo totale dei semilavorati per quello specifico Ordine e va ad aggiornare la TOrdini compilando il campo CostoTotaleSemilavorati
6. **Calcolo del costo Totale del costo Unitario per un prodotto finito**: Input ArticoloID (prodotto finito 1 o 2) :  la WebAPI calcola il costo totale dei semilavorati per quello specifico prodotto finito e va ad aggiornare la TArticoli compilando il campo CostoUnitario solo per quell’articoloID (1 oppure 2) .

**Attenzione**:  ogni chiamata deve verificare se il valore in input ha senso (ArticoloID esistente, OrdineID esistente etc…) e deve rispondere con  un messaggio coerente all’utente.