Ein Werkstattmitarbeiter scannt einen Barcode. Nachdem das Material des gescannten Barcodes angezeigt wurde, wird das Material gezählt und ein Bestand eingegeben und bestätigt. Draufhin wird der **Bestand übermittelt** (vom Scanner an den Server):
`sendeBestand(kennung : string, menge : int)`
Intern rechnet der Werkstattserver nun den Bedarf aus und fügt ihn der Anforderungsliste hinzu.

Ein Werkstattleiter **trägt Sonderbedarf ein**, welcher aus Materialtyp, Anzahl und Begründung besteht. Der Eintrag wird von der GUI an den Werkstattserver gesendet:
`sendeSonderbedarf(material : Materialeintrag, begründung : string)`
Der Sonderbedarf wird dort für spätere Verarbeitung gespeichert.

Der Werkstattleiter **ruft die Liste der Vorratsbehältnisse auf** (GUI holt diese von Server):
`holeBehältnisse() : Vorratsbehältnis[]`
Er wählt ein Vorratsbehältnis aus und **ändert Sollbestand und/oder Material** oder er **legt einen neuen Eintrag an**, wobei Kennung, Materialtyp und Sollbestand eingegeben werden müssen. Für beide Aktionen wird die gleiche Funktion genutzt (GUI übermittelt an Server):
`aktualisiereBehältnis(behältnis : Vorratsbehältnis)`
Wenn ein Behältnis mit noch nicht existenter Kennung „aktualisiert“ wird, erstellt der Server einen neuen Eintrag.

Ein Lagerist nutzt die LagerGUI, welche die **erfüllbaren Lieferaufträge anfordert** (vom Lagerserver):
`holeLieferaufträge() : Lieferauftrag[]`
Der Server übernimmt dabei die Umrechnung von Stückzahl in Verpackungseinheiten. Der Lagerist kann eine Anforderung auswählen (nach Wunsch auch ausdrucken) und **bestätigen**.
`bestätigeLieferauftrag(auftrag : Lieferauftrag)`
Der Server löscht daraufhin den Lieferauftrag und reduziert den Materialbestand.

Ein Lagerist betrachtet alle Bestellungen. (GUI holt Bestellungsliste von Lagerserver):
`holeOffeneBestellungen() : GESBestellbestätigung[]`
Er wählt eine Bestellung aus und kann über die GUI dann Vermerke bezüglich gelieferter Anzahl und Zustand eingeben. Dann werden diese **Vermerke übermittelt** (von der GUI an den Server)
`dokumentiereWareneingang(lieferung : Lieferung)`
Der Server ermittelt daraus den neuen Warenbestand und konvertiert ausstehende Anforderungen von der Warteliste in Lieferaufträge, die dann in Use Case 8 abgearbeitet werden können

Ein Lagerist betrachtet über die GUI eine Liste der Großhändler (GUI **holt Großohändlerliste** von Lagerserver).
`holeGroßhändlerliste() : Großhändlerkontakt[]`
Er kann einen **Großhändler löschen** oder einen **Großhändler aktualisieren bzw. neuerstellen**, wobei der Name eines bestehenden Großhändlers nicht geändert werden kann und zum Erstellen eines neuen Großhändlers einfach ein noch nicht vorhandener Name eingegeben wird. Eine entsprechende Anfrage wird von der GUI an den Lagerserver gesendet:
`löscheGroßhändlerkontakt(großhändler : Großhändlerkontakt)`
`aktualisiereGroßhändlerkontakt(großhändler : Großhändlerkontakt)`
Ein Großhändlerkontakt Eintrag beinhaltet Name, Kontakt, Kontodaten, IP-Adresse und die Artikelbezeichnungen.

Die LagerGUI **fragt beim Lagerserver an, ob eine dringende Nachbestellung erforderlich ist** und holt wenn ja (oder wenn Montag ist) **die Bedarfsliste von Lagerserver** und zeigt sie dem Lagerist:

`dringendeNachbestellungErforderlich() : boolean`
`holeBedarf() : Materialliste`
Die Materialliste kann ausgedruckt werden.

Ein Einkaufsmanager kann in der Lager-GUI eine **Liste aller gelagerten Materialien aufrufen** (beinhaltet jeweiligen Sollbestand):
`holeVorratsliste() : Materialvorrat[]`
Für alle Materialien kann er einen **Sollbestand festlegen**, der dann an den Server übertragen wird. Es wird dabei der gesamte Materialeintrag übertragen, wobei der Server das geänderte Material anhand des Typs identifiziert. Dass nur der Sollbestand, nicht aber z.B. die Packungsgröße geändert werden kann, ist Aufgabe der GUI:
`aktualisiereMaterialvorrat(geänderter_Materialvorrat : Materialvorrat)`