* Geeignete Kameras
  + Im Lager:
    - Unifi theta
  + Nicht im Lager
* Systeme
  + Unifi Theta Kamera
    - Detection Zones
    - Border Lines (Ein- Ausgang erkennen)
    - RTSP-Stream
    - Tensor Flow KI
  + Rasberri Pi KI (<https://tutorials-raspberrypi.de/raspberry-pi-objekterkennung-mittels-tensorflow-und-kamera/?utm_content=cmp-true>)
  + https://www.youtube.com/watch?v=-9cjfw6W\_EU&t=6s
* **Kommunikationsfluss zwischen den Komponenten**
  + **Unifi-Kamera** erfasst ein Bild, wenn ein neues Objekt erkannt wird.
  + Das Bild wird an den **Image Processing Service** im Backend gesendet.
  + Der **Image Processing Service** leitet das Bild an den **TensorFlow-KI-Server** weiter.
  + Der **TensorFlow-KI-Server** analysiert das Bild und identifiziert das Objekt (z. B. ein Fahrrad).
  + Die Erkennungsdaten werden zurück an das **Backend** gesendet, das den **Object Recognition Service** verwendet, um die Informationen in die **Datenbank** zu speichern.
  + Der **Storage Management Service** im Backend ermittelt auf Basis der Objektdaten den passenden Lagerort und weist diesen zu.
  + Die neuen Lagerinformationen werden in der **Datenbank** gespeichert.
  + Das **Frontend** ruft periodisch die Daten vom Backend ab oder erhält diese über Websockets und zeigt die aktualisierten Informationen dem Benutzer an.
* **Besondere Überlegungen**
  + **Skalierbarkeit**: Um Lastspitzen zu bewältigen, könnte der TensorFlow-KI-Server in einer Kubernetes-Umgebung bereitgestellt werden, um eine automatische Skalierung zu ermöglichen.
  + **Echtzeitverarbeitung**: Je nach Leistung der Hardware kann eine asynchrone Verarbeitung implementiert werden, um die Reaktionszeit zu verbessern.
  + **Sicherheit**: Implementierung einer Authentifizierungs- und Autorisierungsebene mittels **JWT-Token** (JSON Web Tokens) oder **OAuth 2.0**.
  + **Fehler-Handling**: Detaillierte Fehlerprotokollierung und Benachrichtigungssysteme für Kameraausfälle oder Erkennungsfehler.