Equipment History manager

listens to:

- order.updated
- order.finished
- agent.registered
- agent.unregistered
- agent.enteredFaultState

Has its own datamodel that is exclusively used to render a ordered history of all events happening.
Index on the agentId

EquipmentController
Handles creating orders and emiting created events

Holds references to all available agents, and their latest emitted state.

Agent[] agents - used to be able to display towards the ui

Listens to:
- order.updated
- order.finished
- agent.registered
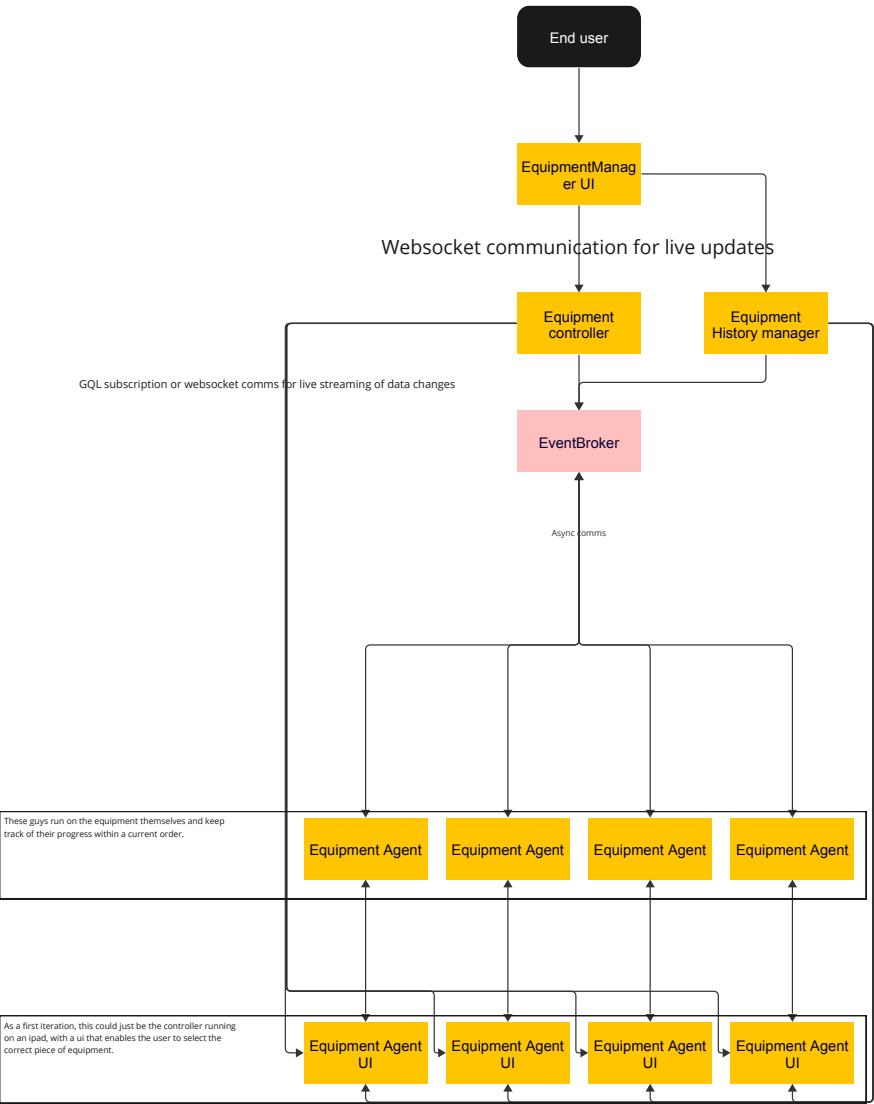- agent.unregistered
- agent.enteredFaultState

Updates the agent state accordingly

EquipmentAgent ui

A ui that hooks into the EquipmentManage API

To allow for changing of states locally and for registering/derigistering as well as allowing to reset the equipment to exit faulty state

Locally an employee will also be able to view the equipment history.

Relevant order Events:
order.created
order.updated
order.finished

Common data for all order.* events:
uuid OrderId
uuid equipmentAgent

Agents are stateful:

Agents listen on
- order.created

Agents emit
- order.updated
- order.finished events

Allowed states
- HandlingOrder
- FinishingOrder
- Idle
- Fault - Wont deal with this state

End user

EquipmentManager UI

Websocket communication for live updates

Equipment controller

Equipment History manager

GQL subscription or websocket comms for live streaming of data changes

EventBroker

Async comms

These guys run on the equipment themselves and keep track of their progress within a current order.

Equipment Agent   Equipment Agent   Equipment Agent   Equipment Agent

As a first iteration, this could just be the controller running on an ipad, with a ui that enables the user to select the correct piece of equipment.

Equipment Agent UI   Equipment Agent UI   Equipment Agent UI   Equipment Agent UI