DHL Smart Automated Warehouse System

JAVA based software simulation

Introduction

DHL operates advanced Smart Warehouses and Distribution Centers around the world, where autonomous robots, conveyor systems, and automated storage units handle parcel sorting, storage, and dispatch.

Our system will simulate a miniature DHL automated warehouse — a digital environment where robots move packages between storage zones, manage charging operations, and coordinate tasks such as loading, unloading, transferring, and dispatching.

Typical DHL workflow

01

Incoming Truck deliver parcels

02

Robots collect parcels from unloading dock

03

Storage Manger decides where to store parcels

04

Order for dispatch triggers.

05

Robots retrieve parcel & deliver to loading dock.

When robots' battery levels drop below 10%, they automatically navigate to the Charging Station.

The HMI Dashboard shows live data about robot status, task queue, and operational logs.

Key Components of the System

01 **Task Management System**

Prioritizes delivery and package handling

02 **Storage Management System**

Keeps track of package location & movement

03 **Charging Station**

For robot maintinence & charging

04 **Human Machine Interface**

To visualize, control & monitor operations

05 **Storage Equipment**

Racks & bins where packages are stored

06 **Logging & Testing**

Exception handling & unit tests for reliability

Requirements Document

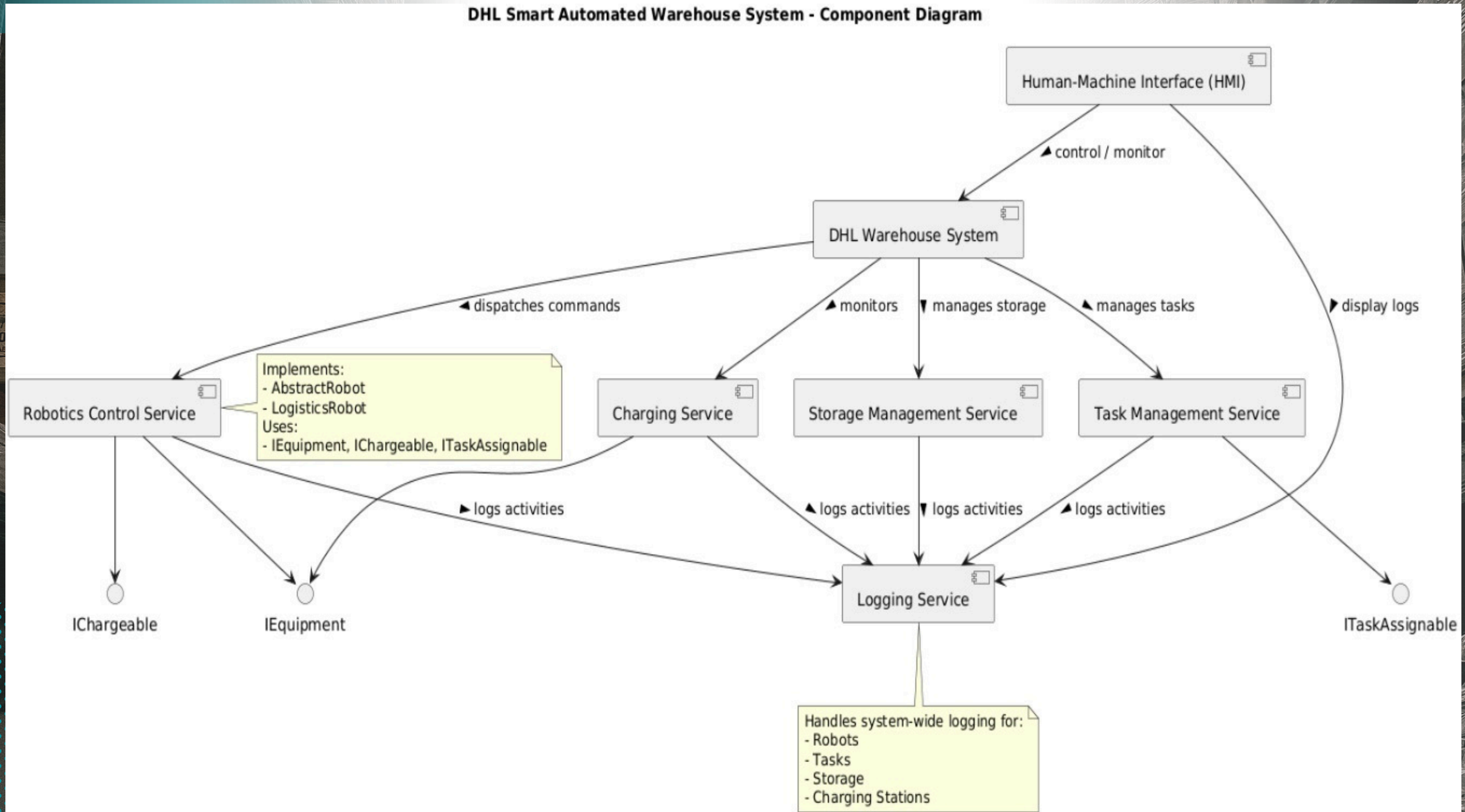
Functional requirments and constraints are provided in a separate file in the capstone repository as a requirements document.

Requirements Document
1. System Overview
- The system simulates a DHL smart automated warehouse.
- Core components include robots, storage systems, task management, charging stations, HMI, ar
a centralized logging system.
2. Functional Requirements
2.1 Equipment
- All equipment must provide a unique ID and a status string.
2.2 Robots
- Robots must implement task assignment and charging functionality.
- Robots must maintain ID, battery level, current task, and operational status.
- Robots must move to locations, execute tasks, pick parcels, deliver parcels, and go to charging
stations.
- Robot actions must be logged.
2.3 Storage System
- StorageManager manages racks and parcel locations.
- It must store and retrieve parcels and find available slots.
- StorageRack maintains slots and supports adding/removing parcels.
- StorageSlot tracks occupancy.
2.4 Task Management
- Supported task types: LOAD, UNLOAD, TRANSFER, CHARGE.
- Each task has ID, type, and status.
- TaskManager creates, assigns, and updates tasks; manages queue; creates dock tasks.
- Robots execute LoadTask, UnloadTask, ChargeTask.
2.5 Charging System
- ChargingStation has ID, availability, occupant.
- It must start/stop robot charging.
- Charging events must be logged.
2.6 DHL Warehouse System
- Maintains StorageManager, TaskManager, robots, and charging stations.
- Provides start, stop, and report generation.
2.7 HMI
- Builds UI, updates UI, refreshes system state, displays logs.
3. Logging Requirements
- Central LogManager logs system, robot, charging, and storage events.
- Must support listing, reading, deleting, moving, and archiving logs.
- Must close log writers on shutdown

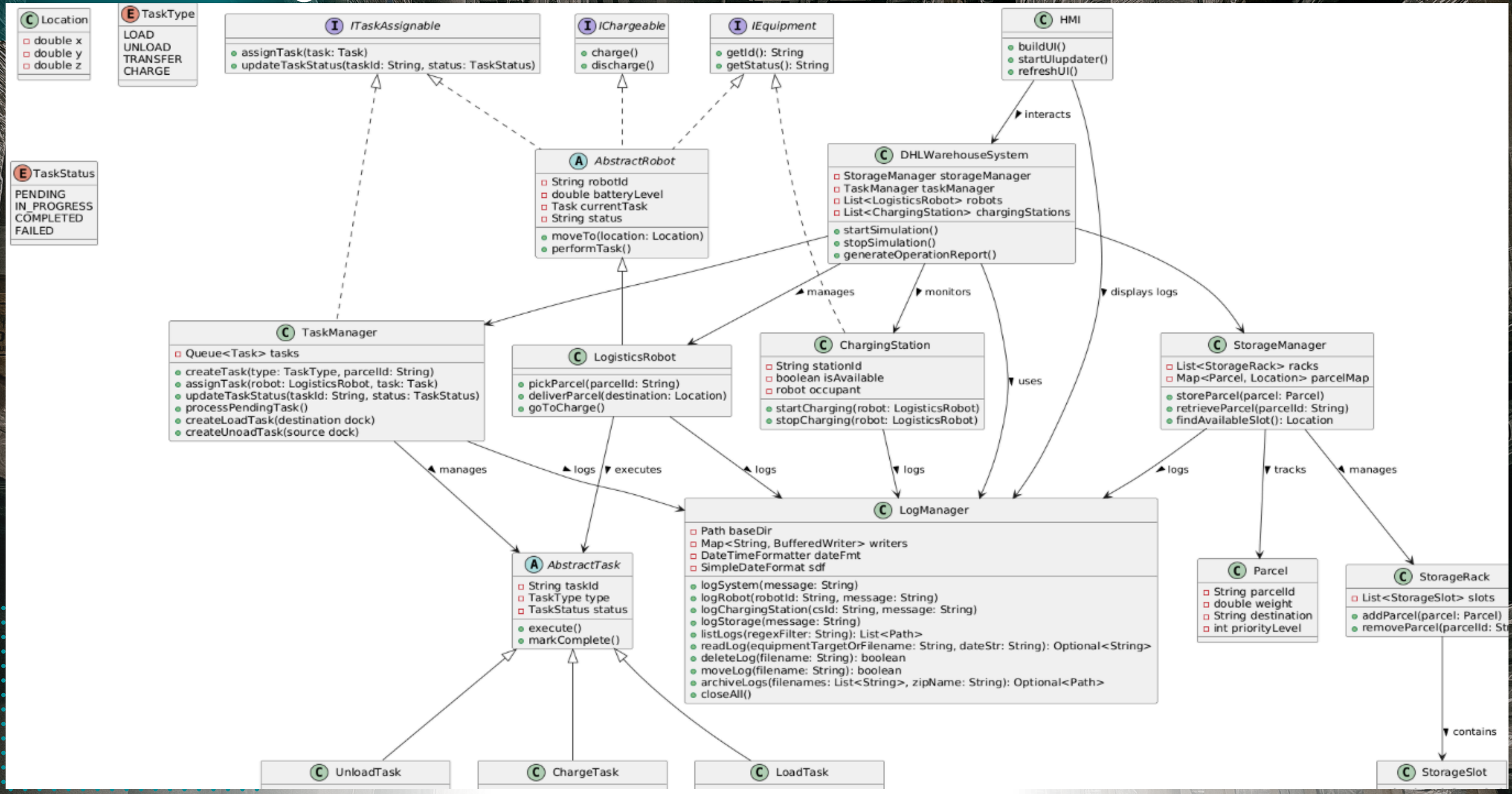
Task Distribution

Muhammad Mohsin Abbasi	HMI, Log Management, Graphics of simulation and overall integration of the system
Minhaj Jamraiz Abbasi	Task Management system Implementing the classes and subclasses related to Task Management
Wajihah Kainat	Storage Management Implementation of Storage Management class and its subclasses including integration into overall system
Asaad Alawy	Robot Management Implementation related to AGVs and it management

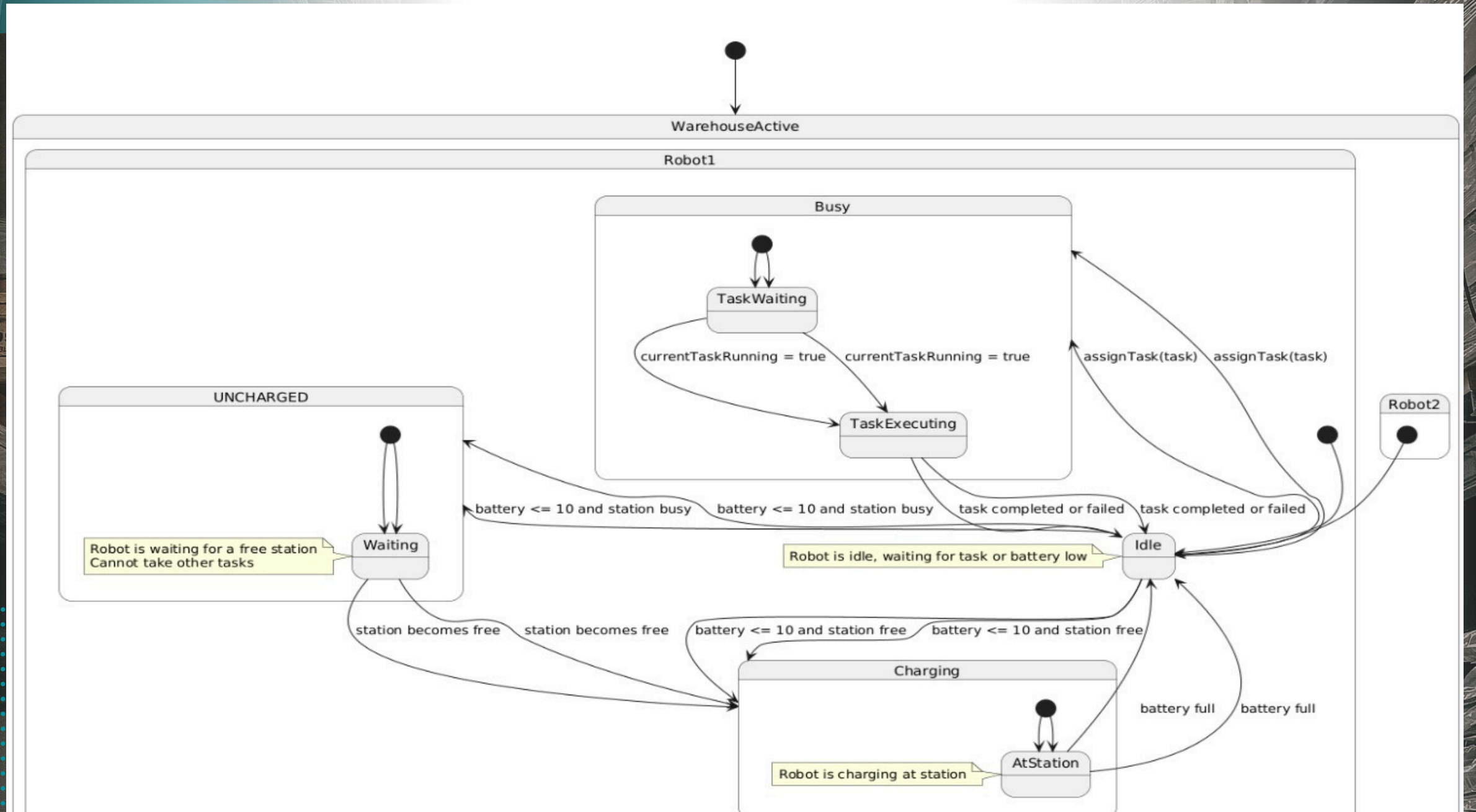
Component diagram



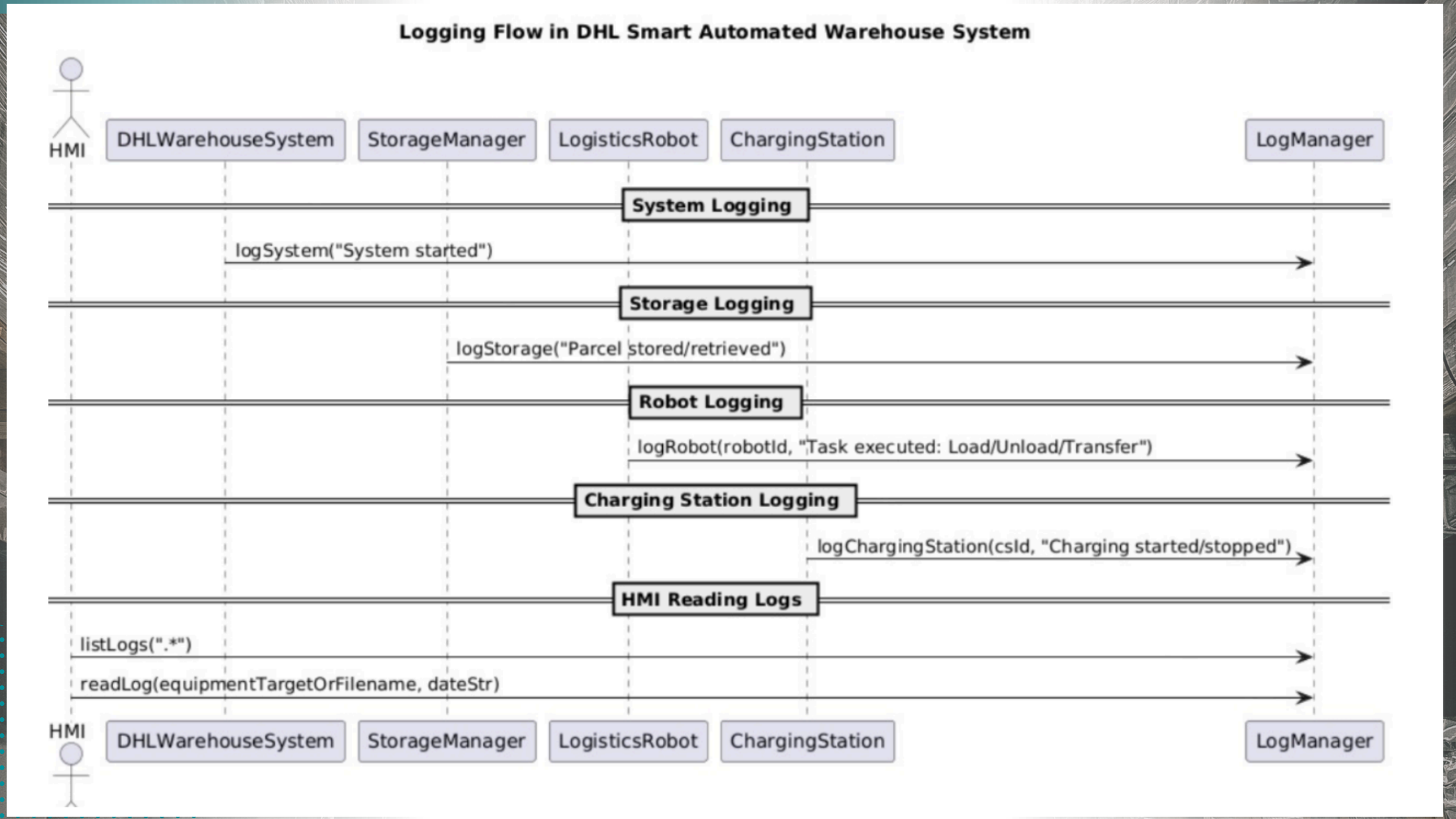
UML class diagram



Concurrency (State Diagram of AGVs)



Management I/O



Simulation

Implemented with JAVAFX for GUI and Timeline simulation.

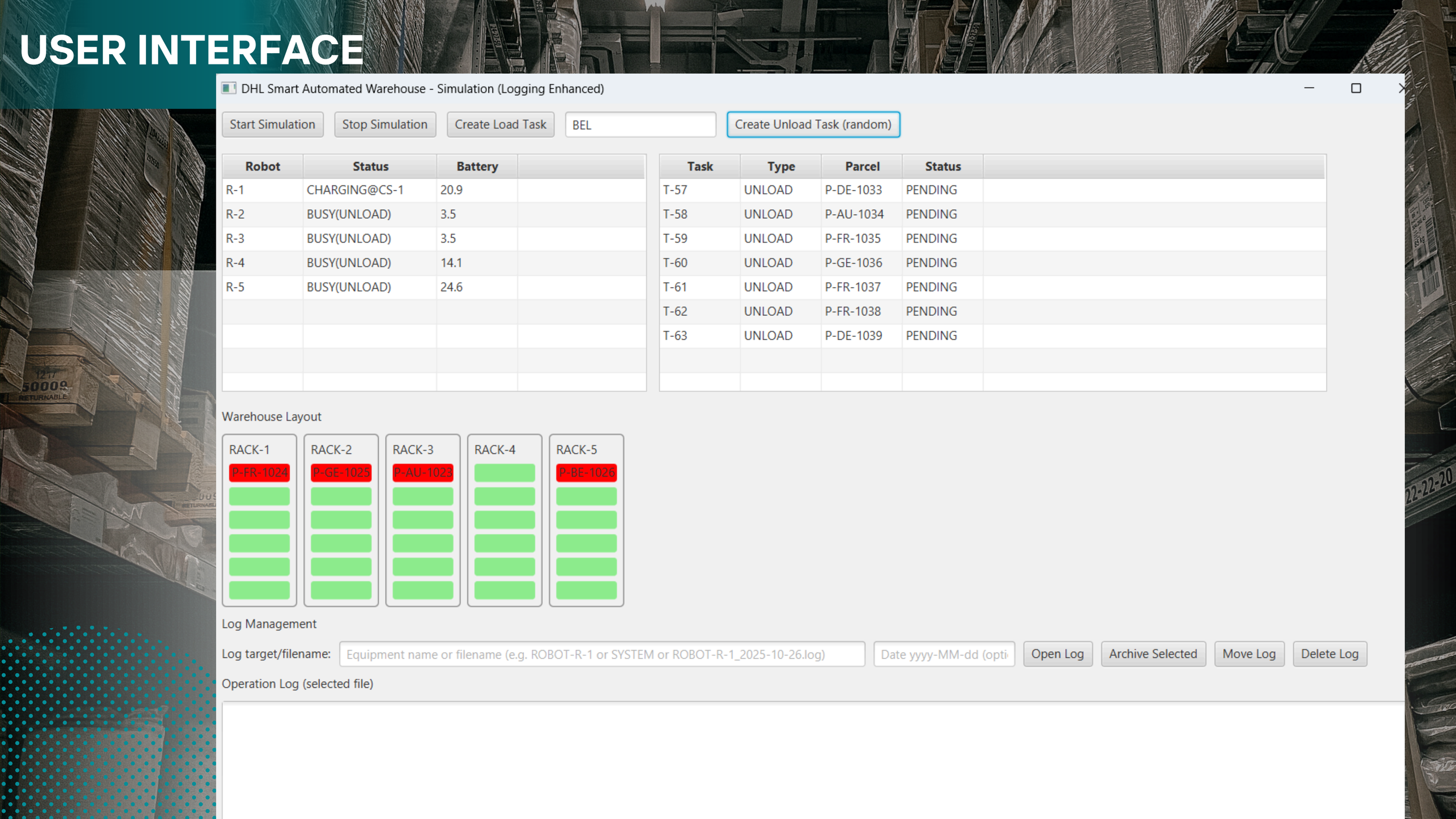
- Real-time, continuous simulation loop
- Decoupled UI and logic for smooth updates

Simulation runs continuously with cycles(ticks), updating warehouse objects.

GUI updated at regular intervals to reflect current system state.

Logs capture actions allowing post simulation analyses

USER INTERFACE




THANK
YOU!