# **Report**

The project is split in 3 main fields which are connected through an mqtt-broker. These 3 fields are the webbots visualization of the robots, the server which calculates all the logic and a website user interface.

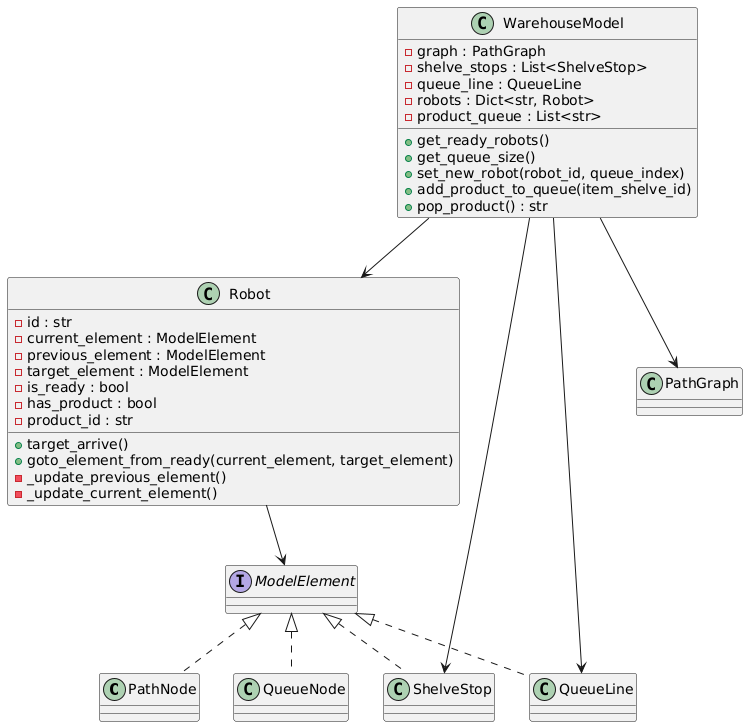
Server:

Figure 1 represents the models that the server initiates and their connections to each other. The WarehouseModel has all the necessary information to guide the Robot instances to their target place in the premade graph.

The scheduling system (figure 2) coordinates the movements of robots in a warehouse, with various components working together:

1. **Scheduler:** Manages task assignment to robots and ensures they perform their work efficiently. It tracks which robots are ready for their next task and whether there are products in the queue.
2. **QueueManager:** Manages robots in the queue, moves them to free positions, and ensures robots drop off products when needed.
3. **TaskManager:** Assigns specific tasks to robots and determines the path to pick up products and return to the queue based on robot status.
4. **PathPlanner:** Calculates the shortest paths for robots to move from one location to another within the warehouse.
5. **ReservationManager:** Ensures robots don’t have conflicts when using paths or nodes by reserving them.

Figure 1: UML diagram server model

1. **RobotCommander:** Sends commands to robots to move, pick up or drop off products, and receives updates on their status.

Together, these components ensure efficient and smooth robot operation in the warehouse, preventing delays and collisions.

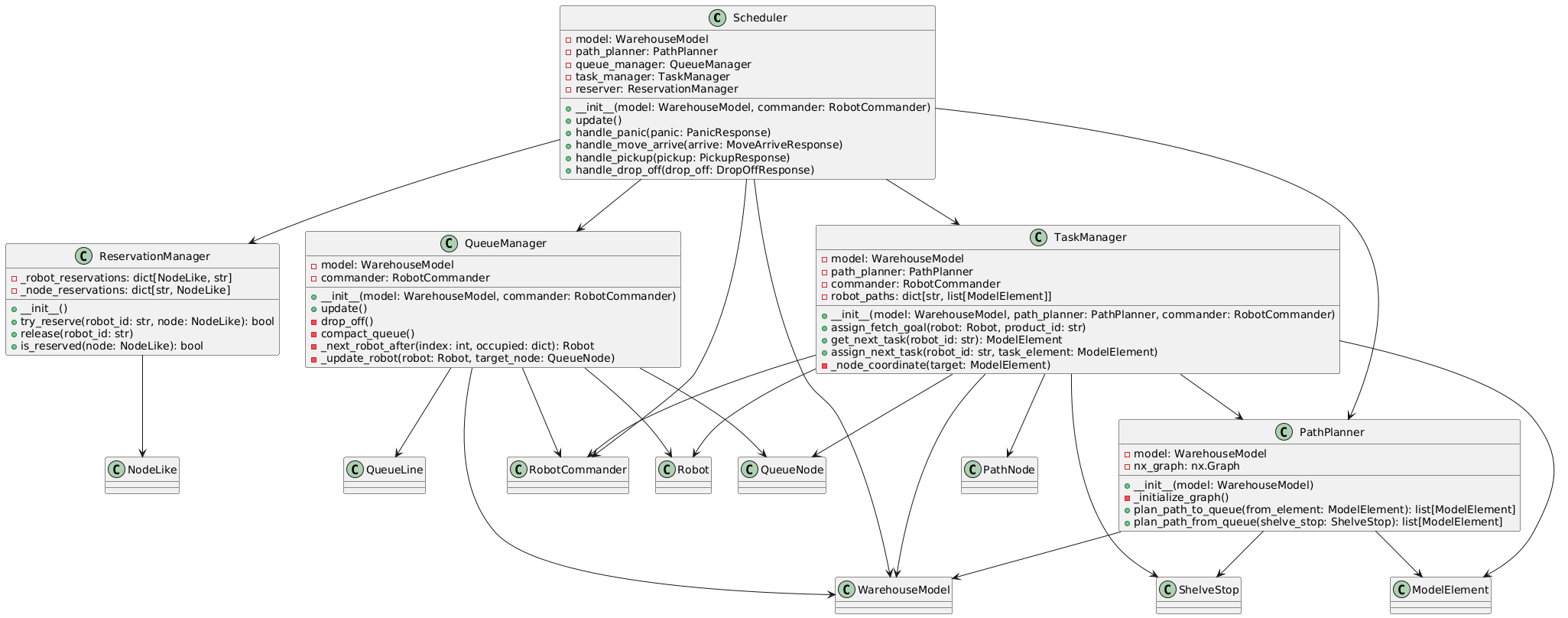


Figure 2: UML diagram scheduling algorithm

Afbeelding met tekst, schermopname

Door AI gegenereerde inhoud is mogelijk onjuist.The connection robot <-> server is completely managed by the RobotCommander class. This class connects to the mqtt-broker, subscribes to the relevant topics to communicate with the robots and sends the correct messages through the correct topics to give commands to the robots.

The added code shows the different commands the RobotCommander can send to the robots and what kind of information every command contains. Every command has a different topic to prevent intervention between different instructions. All commands are processed with the from\_dict() and to\_dict() functions make the dataclasses more compatible with other procedures.

Afbeelding met tekst, schermopname

Door AI gegenereerde inhoud is mogelijk onjuist.

Afbeelding met Schaalmodel, schermopname

Door AI gegenereerde inhoud is mogelijk onjuist.Webots:

The depot made in webots has a few key features which contribute to the correct functioning of the robots. The small black dots on the ground are placed on the nodes of the graph. These will be used for correction if the robot stops on them, after driving to a node, the robot will check if it has a black dot right beneath itself using visual computing. If not, the robot corrects itself until the dot is right beneath it. By doing this, the robot removes slight differences and eventually prevents a significant error.

The robot consist of a main platform (green square) with 4 omniwheels, this design was made by MasdikaAliman (<https://github.com/MasdikaAliman/Kinematic-Omniwheels-in-Webots>). A lidar sensor was added on top of the robot which detect the nearest object and prevents collisions. A color sensor was added beneath the robot to detect the black dots used for correction.

Afbeelding met schermopname, rood, Karmijn

Door AI gegenereerde inhoud is mogelijk onjuist.

Afbeelding met tekst, schermopname, software

Door AI gegenereerde inhoud is mogelijk onjuist.The robot controllers connect to the mqtt-broker with the relevant topics so they can receive commands given by the RobotCommander. The movements by the omniwheels are calculated using a special kinematic file made by MasdikaAliman. Using the omniwheels, the robot does not need to turn around which removes the risks of turning short or long. In other words, the robot can drive straight ahead, sideways and even diagonally.

When the robot does not detect a black dot underneath itself, a panic message is send to the broker so the server knows something is wrong. Same when another object is within 0,5m of the robot.