Simulator für Gewächshäuser

Der Simulator soll 4 Gewächshäuser mit je 8 Pflanzentischen (jeder 1,2 m x 8 m) und unterschiedlichen Pflanzen simulieren.

Die Simulation soll einen Zeitablauf simulieren, der im Betriebszustand analog zur Echtzeit läuft (so dass eine Bedienung über eine GUI Sinn ergibt), sich aber auch zu Testzwecken in einen Modus „schneller Vorlauf" versetzen lassen.

In den Gewächshäusern existieren Sensoren und Aktoren, um das Pflanzenwohlergehen zu kontrollieren und zu beeinflussen.

Name (topic)	Beschreibung	Datentyp	Wertebereich	Einheiten
Sensoren	*Liefert den Zustand im Gewächshaus*			
Temperatur (temp)	Temperatur pro Tisch	Int32Array	in Celcius	-20 - +80
Bodenfeuchte (smoist)	Mittlere Pflanzerdefeuchtigkeit pro Tisch	Int32Array	0-100	rF (rel. Feuchte)
Bodendünger (sferti)	Düngeranteil in der Pflanzenerde	Int32Array	0-100	%
Pflanzengröße (wachst)	Wachstumszustand der Pflanzen pro Tisch (alte Pflanzen)	Int32Array	2-200	cm
Licht (light)	Sonneneinstrahlung (Licht von außen) pro Halle	Int32Array	0-1500	Lumen
Luftfeuchte (humid)	Mittlere relative Feuchtigkeit pro Halle	Int32Array	0-100	%
Aktoren	*Verändere damit den Zustand*	–	–	
Beleuchtung (led)	Künstliche Pflanzenbeleuchung pro Tisch	Int32	0-2000	Watt
Bewässerung (wpump)	Bewässerung pro Tisch ein/aus	BoolArray	True-False	
Düngung (fertil)	Düngungszugabe mit der Bewässerung	BoolArray	True-False	
Beschattung (shading)	Beschattung des Gewächshauses durch Jalousien	Int32	0, 25, 50, 75, 100	%
Lüftung (fan)	Gewächshausbelüftung ein/aus	Bool	True-False	
Allgemein	*Abtastfrequenz*	–	–	
Frequenz (freq)	Sensormessungen (Datenerfassung)	Int32	5-120	Minuten

Die Daten werden in einer Json-Struktur (MQTT-Style) geliefert bzw. erwartet.

Der Entwurf der Software muss die folgenden Eigenschaften berücksichtigen:

- Pro Tisch ist die Anzahl Pflanzen, die Temperatur, die Bodenfeuchte, die Luftfeuchte und das Wachstumsstadium zu speichern, in regelmäßigen Abständen werden die Daten pro Tisch in einer NoSQL-DB gespeichert.
- Pro Halle wird die Beleuchtung und die mittlere Luftfeuchtigkeit gemessen und auch in der NoSQL-DB gespeichert.
- Die Bewässerung und die Tischbeleuchtung können pro Tisch gesteuert werden
- Die Abschattung und die Belüftung können pro Halle gesteuert werden

- Durch Bewässerung, Belüftung und Beleuchtung kann das Pflanzenwachstum beeinflusst werden.
- Benötigen die Pflanzen Dünger kann dies einfach mit der Bewässerung zugegeben werden. Gedüngt wird nur, wenn das Wasser läuft. Während der Bewässerung kann die Düngung zugeschaltet werden. Nur Düngung einschalten hat keine Wirkung.
- Die Simulation arbeitet mit Zeitschritten, die Pflanzen wachsen, aber auch austrocknen lässt. Um dies zu verhindern, muss die Pflanze bewässert werden. Auch die Beleuchtung und die Temperatur haben Einfluss auf das Wachstum und die Pflanzenentwicklung.
- Um Pflanzen zu verschicken, muss eine Pflanze mindestens 30cm hoch sein. Zudem darf die Bodenfeuchte nicht über 50% liegen (Transportgewichtoptimierung).
- Sind auf einem Tisch nur noch 5% Pflanzen, werden die Pflanzen aus dem Gewächshaus entnommen und anderswo vermarktet. Der Tisch erhält neue Setzlinge mit der Anfangsgröße von 2cm.
- Jede Pflanze hat 15cm x 15cm Platz und startet mit einer Anfangsgröße von 2 cm (ebenso Nachpflanzungen). Damit hat jeder Tisch 60 x 8 = 480 Pflanzen.

Simulationsverhalten

- Die Simulation schreitet im Minutentakt voran. Eine simulierte Minute benötigt 0,1 Sekunden.
- Zu Beginn der Simulation ist jeder Tisch voll mit Jungpflanzen (siehe oben).
- Die Pflanzen wachsen zwischen +5 und +40 °C. Optimales Wachstum findet bei rund 22 °C statt. Steigt die Temperatur auf über 60°, sterben die Pflanzen ab.
- Die ideale Bodenfeuchte liegt zwischen 50 bis 80% relativer Bodenfeuchte.
- Pro Minute Bewässerung nimmt die Bodenfeuchte um 5% zu.
- Nach 5 Minuten Düngezugabe sind 100% Dünger pro Pflanze erreicht. Überdüngung sorgt für ein Absterben der Pflanzen. Der Dünger baut sich alle 10 Minuten um 1% ab.
- Die Bodenfeuchte verringert sich in Abhängigkeit der Temperatur am Tisch und der Luftfeuchtigkeit in der Halle. Bei über 80% rel. Luftfeuchte (LF) trocknet der Boden der Pflanzen nicht aus. Darunter trocknet der Boden um ein 1/10 der Differenz zwischen Bodenfeuchte (BF) und Luftfeuchte pro Stunde. Beispiel: BF 70%, LF 40%: Abnahme der BF 3% pro Stunde.
- Die Luftfeuchte kann nicht über 85% steigen, da dann die übersteigende Feuchte aus der Halle entweicht.
- Pro 500 Lumen Lichteinstrahlung erhöht sich die Hallentemperatur um 3° (Wärmeanteil im Sonnenlicht).
- Die Jalousien reduzieren die äußere Lichteinstrahlung um die prozentualen Stufen.
- Die Lüftung ist erforderlich, da die Pflanzen CO2 abbauen und Sauerstoff erzeugen. Die Lüftung tauscht die Luft komplett gegen die Außenluft auf, wenn sie 30 Minuten läuft. Gleichzeitig kühlt sie aber auch die Halle. Nach 30 Minuten hat die Halle die Außentemperatur und die -luftfeuchte.

Ablauf: (pro Zeitschritt)

- Simulation der Jahreszeiten (Beginn Frühling), dann des Monats und den Tag. Über den Tag Temperatur, Luftfeuchte, Sonne.
- Dann Aktualisierung des Hallenzustands
- Aktualisierung der Tische
- Speicherung der Daten je nach Frequenzvorgaben (Start: alle 5 Minuten)

Prerequisiten: NodeJs muss installiert sein. Der Rechner muss Zugang zum Internet haben, um Module herunterzuladen.

Aufgaben (ToDo):

1. Entpacken des unfertigen Codes (greenhouse.zip). Die Struktur besteht aus dem Ordner `server` und den beiden Unterordnern `src` und `node_modules`. Der Ordner node_modules ist leer. Der Ordner frontend kann unächst ignoriert werden.
2. Öffnet ein Terminal und wechselt in den Ordner `server`. Dort gibt es die Datei `package.json`. Startet im Terminal den Befehl `npm install`. Danach sind alle Pakete installiert (kann in `node_modules` kontrolliert werden).
3. Testet den Server mit dem Befehl `npm run dev` (siehe dazu auch die Befehle in start.bat). Wie man sehen kann, findet keine echte Simulation statt.
4. Implementiert in den beiden Funktionen simulateAllGreenhouses und updateTableData eine näherungsweise realistische Simulation anhand der oben gegebenen Bedingungen. Die Verwendung von Internetquellen und KI-Hilfestellungen ist ausdrücklich erwünscht. Die vorhandenen API-Endpunkte sind:
   `GET /api/greenhouses` - Alle Gewächshäuser
   `GET /api/greenhouses/:id/sensors` - Spezifisches Gewächshaus
   `GET /api/tables/:greenhouseId/:tableId` - Spezifischer Tisch
   `GET /api/health` - Health Check
   `WebSocket ws://localhost:3001` - Live Updates (optional)
   Ergänzt die REST-API-Endpunkte um alle Befehle, die erforderlich sind, um die Gewächshäuser zu steuern.
5. Dabei wird schnell offensichtlich, dass man sich eine Teststrategie überlegen muss, da die Simulation zustandsabhängig ist.
6. Testet die API-Endpunkte zunächst mit Postman (oder passende Alternative).
7. Entwerft Tests zu der Funktion updateTableData. Plant dazu zunächst, welche Art(en) von Tests ihr braucht.

Deckt mindestens den folgenden Test-Umfang ab:

- *Initialisierung: Constructor, Environment Data laden*
- *Simulation: Temperatur, Licht, Luftfeuchtigkeit, Pflanzenwachstum, Bodenbedingungen*
- *Zeitmanagement: Minuten, Stunden, Tage*
- *Integration: Komplette Tages-/Wochenzyklen*

8. Fasst die von euch erstellten API-Endpunkte zu einer Mini-API-Dokumentation zusammen, aus der zu entnehmen ist, welcher Befehl (GET, POST, ..) mit welchen Parametern verfügbar ist und was er bewirken soll.