AssetRoutes.js

Die Datei assetRoutes.js **definiert die Endpunkte (Routen)** für die Asset-API.  
Sie sorgt dafür, dass die richtigen **Controller-Funktionen** aufgerufen werden, wenn Anfragen an die API gestellt werden.

**Ablauf des Codes (assetRoutes.js)**

1️. i**mport der benötigten Module**

import express from "express";

import { getAssets, createAsset, updateAsset, deleteAsset, getApiStatus } from "../controllers/assetController.js";

→ **express** wird genutzt, um die API-Routen zu erstellen.  
→ **getAssets, createAsset, updateAsset, deleteAsset, getApiStatus** werden aus assetController.js importiert.  
→ Diese Funktionen sind für das **Abrufen, Erstellen, Aktualisieren und Löschen** von Assets zuständig.

2️. **Router-Objekt erstellen**

const router = express.Router();

→ Erstellt einen **neuen Express-Router**, der API-Anfragen verarbeitet.

3️. **Routen definieren**

router.get("/", getAssets); // Alle Assets abrufen

router.post("/", createAsset); // Neues Asset hinzufügen

router.put("/:id", updateAsset); // Asset mit bestimmter ID aktualisieren

router.delete("/:id", deleteAsset); // Asset mit bestimmter ID löschen

router.get("/status", getApiStatus); // API-Status abfragen

→ **router.get("/")** ruft getAssets auf, um alle gespeicherten Assets abzurufen.  
→ **router.post("/")** ruft createAsset auf, um ein neues Asset zu erstellen.  
→ **router.put("/:id")** aktualisiert ein Asset anhand der **ID**, die in der URL angegeben wird.  
→ **router.delete("/:id")** löscht ein Asset anhand der **ID**.  
→ **router.get("/status")** gibt den API-Status zurück, um zu prüfen, ob die API läuft.

4️. **Router exportieren**

export default router;

→ Der **Router wird exportiert**, damit er in app.js eingebunden werden kann.

**📊 Übersicht der API-Routen**

|  |  |  |  |
| --- | --- | --- | --- |
| **Methode** | **Route** | **Funktion** | **Beschreibung** |
| **GET** | /api/assets | getAssets | Alle Assets abrufen |
| **POST** | /api/assets | createAsset | Neues Asset erstellen |
| **PUT** | /api/assets/:id | updateAsset | Asset mit ID aktualisieren |
| **DELETE** | /api/assets/:id | deleteAsset | Asset mit ID löschen |
| **GET** | /api/assets/status | getApiStatus | API-Status prüfen |

**🛠️ Warum ist dieser Code wichtig?**

✅ **Klare Trennung** zwischen API-Routen und der Logik in assetController.js.  
✅ **Einfache Wartung**: Falls sich eine API-Route ändert, muss nur diese Datei angepasst werden.  
✅ **Saubere Struktur**, da alle **Routen an einem Ort** definiert sind.