Kompetenz-Baumstruktur: Bottom-Up Lernpfad für verteilte Systeme im PDAL (Granularisiert & Erweitert)

Root-Kompetenz (Das ultimative Ziel):

- **Erfahrungen mit verteilten Anwendungen und Middlewaresystemen sammeln**
    - *Kompetenz: Entwurf & Implementierung von Multi-Service-Architekturen (Ebene 4)*

Individueller Lernpfad: Ihr Weg zum Personal Distributed Applications Lab (PDAL)

Dieser Kompetenzbaum dient als Leitfaden für Ihren individuellen Lernpfad im PDAL-Projekt. Die Module sind vorwiegend als **Selbstlernmodule** konzipiert, die den Lernenden ermöglichen, in Ihrem eigenen Tempo zu arbeiten.

Um eine solide Wissensbasis aufzubauen, empfehlen wir, die **Grundlagenmodule der Ebene 0 und Ebene 1** vollständig zu bearbeiten. Diese Module legen das absolute Fundament für alle weiteren Schritte.

Anschließend können Sie den Lernpfad **flexibel und nach Interessen der Lernenden** gestalten. Bei vielen Modulen handelt es sich um Vertiefungs- und Verfestigungsaufgaben, die dazu dienen, theoretische Inhalte durch praktische Experimente zu untermauern. Zudem können bei Bedarf auch **eigene, themenspezifische Module** in den Lernpfad integriert werden, um sich auf bestimmte Technologien oder Anwendungsfälle zu konzentrieren.

Ziel ist es, Ihnen ein strukturiertes, aber dennoch anpassbares Framework an die Hand zu geben, um Ihre Kompetenzen im Umgang mit verteilten Anwendungen und Middlewaresystemen gezielt aufzubauen und zu vertiefen.

Ebene 4: Aufbau & Test von komplexen verteilten Anwendungen (Ca. 10-25 Stunden pro Lerneinheit)

- **Kompetenz: Entwurf & Implementierung von Multi-Service-Architekturen**
    - **Beschreibung:** Die Fähigkeit, eine mehrschichtige, verteilte Anwendung zu entwerfen, zu implementieren und ihre Interaktionen innerhalb des PDAL zu testen. Dies beinhaltet das Verständnis für Fehlertoleranz und die Dokumentation eigener Projekte.
    - *Voraussetzung: Kompetenz "Analyse & Fehlerbehebung in verteilten Systemen" (Ebene 3)*
    - **Lerneinheiten (Qualifikationen):**
        - **Lerneinheit 4.1: Entwurf einer einfachen Microservices-Anwendung (3-5 Stunden)**
            - **Ziel:** Verstehen der Konzepte von Microservices und Entwerfen einer minimalen, verteilten Anwendung (z.B. ein Frontend, ein Backend-API-Service, ein Daten-Service).

        - **Inhalte:** Einführung in Microservices, Skizzieren der Architektur und Kommunikation zwischen Diensten.
    - **Lerneinheit 4.2: Implementierung des Frontend-Microservice (3-7 Stunden)**
        - **Ziel:** Entwicklung und Bereitstellung eines einfachen Web-Frontends in einem LXC, das mit dem Backend kommuniziert.
        - **Inhalte:** Auswahl einer Technologie (z.B. Python Flask, Node.js Express, einfacher statischer Webserver), Implementierung, Deployment im LXC.
    - **Lerneinheit 4.3: Implementierung des Backend-API-Microservice (3-7 Stunden)**
        - **Ziel:** Entwicklung und Bereitstellung eines **REST-API-Services** in einem LXC, der Daten verarbeitet und ggf. eine Datenbank nutzt.
        - **Inhalte:** REST-API-Design (HTTP-Methoden, Statuscodes), Datenmodellierung, Implementierung der Logik, Deployment im LXC.
    - **Lerneinheit 4.4: Integration und Test des Multi-Service-Szenarios (5-10 Stunden)**
        - **Ziel:** Zusammenspiel aller Microservices testen, End-to-End-Funktionalität überprüfen und grundlegende Fehlerbehebung auf Anwendungsebene.
        - **Inhalte:** Netzwerkkommunikation zwischen Services, Konfiguration von Umgebungsvariablen, Teststrategien.
    - **Lerneinheit 4.5: Grundlagen der Container-Orchestrierung (Optional/Ausblick, 5-10 Stunden)**
        - **Ziel:** Erste Schritte mit Docker und Docker Compose innerhalb eines LXC zur einfachen Orchestrierung mehrerer Service-Container.
        - **Inhalte:** Installation von Docker in einem LXC, Dockerfiles, Docker Compose-Dateien, Starten von Multi-Container-Anwendungen.
    - **Lerneinheit 4.6: Experimente zu Fehlertoleranz & Resilienz (5-10 Stunden)**
        - **Ziel:** Bewusstes Herbeiführen von Fehlern (z.B. Netzwerk trennen, Dienst stoppen) und Analyse der Auswirkungen auf die verteilte Anwendung. Erste Überlegungen zu resilienten Designs.
        - **Inhalte:** Chaos Engineering light, Fehlerprotokollierung, Auswirkungen auf Verfügbarkeit.
    - **Lerneinheit 4.7: Projekt JavaServer-Applikation (3-5 Stunden)**
        - **Ziel:** Erstellen eines LXC mit TomCat-Server und Entwicklung einer JSP im PDAL.
        - **Inhalte:** Installation Apache Tomcat, Entwicklung eines JSP, Deployment JSP auf Tomcat, Implementierung einer Datenbank in JSP.
    - **Lerneinheit 4.8: Projekt-Dokumentation (3-5 Stunden)**
        - **Ziel:** Erstellen einer klaren Dokumentation des Aufbaus, der Konfiguration und der Funktionalität der entwickelten verteilten Anwendung im PDAL.
        - **Inhalte:** Architekturdiagramme, Konfigurationsschritte, Testfälle.

Ebene 3: System-Interaktion & Fehlerbehebungs-Kompetenz (Ca. 5-15 Stunden pro Lerneinheit)

- **Kompetenz: Analyse & Fehlerbehebung in verteilten Systemen**
    - **Beschreibung:** Die Fähigkeit, die Kommunikation und Abhängigkeiten zwischen verschiedenen Diensten zu verstehen und auftretende Probleme systematisch zu identifizieren und zu beheben.

- *Voraussetzung: Kompetenz "Bereitstellung & Konfiguration verteilter Systemkomponenten" (Ebene 2)*

- **Lerneinheiten (Qualifikationen):**

  - **Lerneinheit 3.1: Grundlagen der Netzwerk-Kommunikationsanalyse (5-8 Stunden)**
    - **Ziel:** Erlernen des Einsatzes von `tcpdump` oder `wireshark` zur Paketanalyse zwischen Containern.
    - **Inhalte:** Installation der Tools, Filtern von Traffic, Identifikation von Quell-/Ziel-IPs und Ports, grundlegende Protokollanalyse (HTTP, DNS).
  - **Lerneinheit 3.2: Fortgeschrittene Log-Analyse (3-5 Stunden)**
    - **Ziel:** Effektives Auffinden, Filtern und Interpretieren von System- und Anwendungslogs in Linux-Containern zur Fehlersuche.
    - **Inhalte:** `journalctl` (systemd logs), `grep`, `tail -f`, Log-Rotation, Bedeutung gängiger Log-Meldungen, Debug-Mode.
  - **Lerneinheit 3.3: Analyse von Abhängigkeiten und deren Auswirkungen (3-5 Stunden)**
    - **Ziel:** Erstellen eines Abhängigkeitsdiagramms für eine einfache, verteilte Anwendung und Simulation von Dienstausfällen.
    - **Inhalte:** Service-Maps, Verständnis von Aufrufketten, Auswirkungen von Latenz und Ausfällen.
  - **Lerneinheit 3.4: Ressourcenüberwachung und Performance-Analyse (4-7 Stunden)**
    - **Ziel:** Überwachen von CPU, RAM, Netzwerk-I/O und Festplatten-Nutzung in LXC-Containern zur Identifikation von Performance-Engpässen.
    - **Inhalte:** `htop`, `nmon`, `iostat`, `dstat`, `ss`, Interpretation der Metriken.

Ebene 2: Middleware- und Anwendungsbereitstellungs-Kompetenz (Ca. 5-15 Stunden pro Lerneinheit)

- **Kompetenz: Bereitstellung & Konfiguration verteilter Systemkomponenten**

  - **Beschreibung:** Die Fähigkeit, diverse Middlewaresysteme und Anwendungen (Webserver, Datenbanken, Message Broker etc.) in einzelnen LXC-Containern zu installieren und für die verteilte Kommunikation zu konfigurieren.

  - *Voraussetzung: Kompetenz "Effektives Management von Linux-LXC-Containern" (Ebene 1)*

  - **Lerneinheiten (Qualifikationen):**

    - **Lerneinheit 2.05: Eigene Certificate-Authority anlegen(CA) (2-3 Stunden)**
      - **Ziel:** Eine eigene Zertifikatsverwaltung in das PDAL integrieren und die Verwendung der Zertifikate in unterschiedlichen Servern.
      - **Inhalte:** `apt install openSSL`, Root-Zertifikat erstellen, Server-/Dienstzertifikate erstellen, PrivateKey erstellen und Verteilung an unterschiedliche Stellen (nicht automatisiert).
    - **Lerneinheit 2.1: Webserver Apache2: HTTP und PHP (5-8 Stunden)**
      - **Ziel:** Installation und Grundkonfiguration von Apache2 als Webserver, Bereitstellung von HTTP-Inhalten und Integration von PHP.

    - **Inhalte:** `apt install apache2 php libapache2-mod-php`, Konfigurationsdateien, virtuellen Hosts, `phpinfo()`.
- **Lerneinheit 2.2: MariaDB und phpMyAdmin Integration (5-8 Stunden)**
    - **Ziel:** Installation von MariaDB als Datenbankserver und phpMyAdmin zur Administration. Anbindung an Apache2/PHP.
    - **Inhalte:** `apt install mariadb-server phpmyadmin`, Benutzer- und Rechteverwaltung in MariaDB, phpMyAdmin-Konfiguration, Erstellung einfacher Datenbanken und Tabellen.
- **Lerneinheit 2.3: Eigene Webanwendung (HTML/PHP) mit Datenbankanbindung (5-10 Stunden)**
    - **Ziel:** Entwicklung einer einfachen Webanwendung in HTML/PHP, die Daten in MariaDB speichert und ausliest, über Apache2 zugänglich.
    - **Inhalte:** Grundlegende PHP-Skripte für Datenbankinteraktion, Formularverarbeitung, einfache CRUD-Operationen (Create, Read, Update, Delete).
- **Lerneinheit 2.4: REST-Anwendungen: Eigenentwicklung und externe Nutzung (5-10 Stunden)**
    - **Ziel:** Entwicklung einer einfachen eigenen **REST-API** (z.B. mit Python Flask oder PHP) und Nutzung einer externen REST-API von einer anderen Anwendung/Service.
    - **Inhalte:** HTTP-Methoden (GET, POST, PUT, DELETE), JSON-Datenformat, Implementierung eines API-Endpunkts, `curl` oder Python `requests` für externe API-Calls.
- **Lerneinheit 2.5: MQTT Broker (Mosquitto): Anonym & Authentifiziert (6-10 Stunden)**
    - **Ziel:** Installation und Konfiguration des Mosquitto MQTT Brokers für anonyme und authentifizierte Verbindungen (Benutzer/Passwort).
    - **Inhalte:** `apt install mosquitto`, `mosquitto.conf`, `mosquitto_passwd`, ACLs, Test mit `mosquitto_pub`/`mosquitto_sub`.
- **Lerneinheit 2.5.2: MQTT Client-Implementierung (Python & Java) (8-12 Stunden)**
    - **Ziel:** Entwicklung eigener Publisher- und Subscriber-Anwendungen in Python und Java, die mit dem konfigurierten MQTT Broker kommunizieren.
    - **Inhalte:** Nutzung von MQTT-Bibliotheken (z.B. Paho MQTT für Python/Java), Topics, QoS-Level, Senden/Empfangen von Nachrichten.
- **Lerneinheit 2.5.3: MQTT Broker: TLS-Verschlüsselung (4-6 Stunden)**
    - **Ziel:** Absicherung der MQTT-Kommunikation mit TLS/SSL unter Verwendung eigener Zertifikate (aus Lerneinheit 2.8).
    - **Inhalte:** TLS-Konfiguration in `mosquitto.conf`, Client-Zertifikatsvalidierung.
- **Lerneinheit 2.6: Datenbank PostgrSQL und Datenbankmangementsystem (DBMS) pgadmin4**
    - **Ziel:** Installation von PostgreSQL und pgadmin4 und Anlegen unterschiedlicher Nutzergruppen/-user, Anlegen von Datenbank und Zuweisung von Rechten, Absicherung mit SSL
    - **Inhalte:** `apt install postgres`, Benutzer- und Rechteverwaltung, SSL-Zertifikate, Konfiguration PostgreSQL und pgadmin4.
- **Lerneinheit 2.7: Zertifikatsverwaltung und HTTPS für Apache2 (4-6 Stunden)**
    - **Ziel:** Einrichten einer einfachen lokalen CA in einem Container und Erstellen/Signieren von SSL/TLS-Zertifikaten für Apache2 zur Aktivierung von **HTTPS**.

        - **Inhalte:** OpenSSL-Befehle (CA, Schlüssel, CSR, Signieren), Apache2-SSL-Modul (`a2enmod ssl`), Konfiguration von SSL-Virtual-Hosts.
    - **Lerneinheit 2.8: JupyterLab: Python, Java, R für Datenanalyse & Interaktion (6-10 Stunden)**
        - **Ziel:** Installation und Konfiguration von JupyterLab und Nutzung für interaktive Skripte und Datenanalyse in Python, Java und R.
        - **Inhalte:** `apt install jupyterlab`, Installation von Kernels (Python, IJava, R-Kernel), Einbinden externer Bibliotheken (z.B. `pandas`, `requests`, `paho`), Schreiben einfacher Code-Zellen zur Interaktion mit anderen PDAL-Diensten (z.B. REST-API-Calls, MQTT-Publishing).
    - **Lerneinheit 2.9: Grafana: Monitoring und Visualisierung (8-12 Stunden)**
        - **Ziel:** Installation und Konfiguration von Grafana zur Visualisierung von Daten aus MariaDB oder anderen Quellen (z.B. Prometheus Node Exporter in LXCs).
        - **Inhalte:** `apt install grafana`, Hinzufügen von Datenquellen (MariaDB, Prometheus), Erstellen eigener Dashboards mit verschiedenen Visualisierungen, Abfragen von Daten.

Ebene 1: Betriebssystem- und Container-Management-Kompetenz (Ca. 3-8 Stunden pro Lerneinheit)

- **Kompetenz: Effektives Management von Linux-LXC-Containern**
    - **Beschreibung:** Die Fähigkeit, Ubuntu LTS LXC-Container zu provisionieren, zu konfigurieren und grundlegende Linux-Aufgaben innerhalb der Container auszuführen.
    - *Voraussetzung: Kompetenz "Beherrschen der PDAL-Hardware & Basis-Virtualisierung" (Ebene 0)*
    - **Lerneinheiten (Qualifikationen):**
        - **Lerneinheit 1.1: Erster Ubuntu LTS LXC-Container (2-3 Stunden)**
            - **Ziel:** Erfolgreiches Erstellen eines Ubuntu LTS LXC-Containers aus einem Proxmox-Template und erster Login.
            - **Inhalte:** Auswahl des Templates, Ressourcenzuweisung, Konfiguration des Netzwerks (DHCP), Starten des Containers, Login per Konsole/SSH.
        - **Lerneinheit 1.2: Grundlegende Linux-Shell-Navigation und -Befehle (3-5 Stunden)**
            - **Ziel:** Sicherer Umgang mit grundlegenden Linux-Befehlen für Dateisystem-, Benutzer- und Prozessverwaltung.
            - **Inhalte:** `cd`, `ls`, `mkdir`, `rm`, `cp`, `mv`, `cat`, `more`, `less`, `sudo`, `useradd`, `passwd`, `ps`, `top`, `kill`, `df`, `du`.
        - **Lerneinheit 1.3: Netzwerkkonfiguration und -diagnose im Container (2-4 Stunden)**
            - **Ziel:** Konfiguration statischer IPs im Container und effektive Nutzung von Netzwerkdiagnose-Tools.
            - **Inhalte:** Bearbeiten von `/etc/netplan` oder `/etc/network/interfaces`, `ping`, `ip a`, `ip r`, `netstat -tulnp`, `curl`, `wget`.
        - **Lerneinheit 1.4: Software-Installation und Paketverwaltung (2-3 Stunden)**
            - **Ziel:** Sicherer Umgang mit `apt` zur Installation, Aktualisierung und Entfernung von Softwarepaketen.

        - **Inhalte:** `apt update`, `apt upgrade`, `apt install`, `apt remove`, `apt search`, Hinzufügen von PPAs/Repositories.

Ebene 0: Basis-Infrastruktur-Kompetenz (Das absolute Fundament) (Ca. 1-5 Stunden pro Lerneinheit)

- **Kompetenz: Beherrschen der PDAL-Hardware & Basis-Virtualisierung**
    - **Beschreibung:** Die grundlegende Fähigkeit, die Hardware des Tiny-PCs vorzubereiten, Proxmox zu installieren und die essentielle, isolierte Netzwerkverbindung für das PDAL herzustellen.
    - *Voraussetzung: Keine weiteren Kompetenzen. Dies ist der Startpunkt.*
    - **Lerneinheiten (Qualifikationen):**
        - **Lerneinheit 0.1: Auswahl und Vorbereitung des Tiny-PCs (1-2 Stunden)**
            - **Ziel:** Verständnis der Hardware-Anforderungen und Vorbereitung des Tiny-PCs für die Proxmox-Installation.
            - **Inhalte:** Mindestanforderungen (RAM, CPU, Speicher), BIOS/UEFI-Einstellungen (Virtualisierung aktivieren), Boot-Reihenfolge.
        - **Lerneinheit 0.2: Grundlegende Netzwerkkonfiguration des Host-PCs (Windows) für ICS (2-3 Stunden)**
            - **Ziel:** Korrektes Einrichten von Internet Connection Sharing (ICS) unter Windows, um dem Tiny-PC isolierten Internetzugang zu ermöglichen.
            - **Inhalte:** Installation der USB-Netzwerkkarte, Aktivieren von ICS für die LAN-Verbindung (USB-Karte), Verständnis der resultierenden IP-Adressbereiche (192.168.137.x).
        - **Lerneinheit 0.3: Installation von Proxmox VE (2-4 Stunden)**
            - **Ziel:** Erfolgreiche Installation von Proxmox VE auf dem Tiny-PC als Bare-Metal-Hypervisor.
            - **Inhalte:** Download des Proxmox ISOs, Erstellen eines bootfähigen USB-Sticks, Durchführung der Installation, Erstkonfiguration (IP-Adresse, Root-Passwort).
        - **Lerneinheit 0.4: Erster Zugriff auf die Proxmox Weboberfläche (1-2 Stunden)**
            - **Ziel:** Erfolgreicher Zugriff auf die Proxmox-Verwaltungsoberfläche über den Host-PC und erste Orientierung im Dashboard.
            - **Inhalte:** Eingabe der Proxmox-IP im Browser, Login mit Root-Zugangsdaten, Überblick über die Hauptmenüpunkte.
        - **Lerneinheit 0.5: Einführung in die Proxmox-Speicherverwaltung (1-2 Stunden)**
            - **Ziel:** Verständnis der verschiedenen Speichertypen in Proxmox und Erstellung des ersten Speicherpools für Container-Templates.
            - **Inhalte:** Konzepte von LVM, ZFS, Verzeichnis als Speichertypen, Hinzufügen eines Verzeichnisses, Download des ersten LXC-Templates.

Lizenz