Absolutely. Here's your **step-by-step, systematic plan** for the **Progress Report 1 (due 30 June 2025)** project — structured into *phases*, *key deliverables*, and *dependencies*. This is tailored for both your academic goals and strategic job positioning.

**📘 Progress Report 1 – Master Plan**

**Goal**: Simulate real Swiss train operations in SUMO **without VC**, generate KPIs, and prepare a reproducible framework for future VC integration.

**🔷 PHASE 0 – Environment, Standards, and Structure**

**✅ Objectives**

* ✅ Use only **local files and folders**
* ✅ Follow **professional Python coding standards** (logging, error handling, Google-style docstrings)
* ✅ Use GitHub repo for tracking (sumo-swiss-network-june-2025)
* ✅ Maintain this folder structure:  
  D:/PhD/prog\_report\_2025\_June\_project/

**🔷 PHASE 1 – SUMO Network Reconstruction**

**📌 Goal**

Rebuild the Swiss railway network in SUMO from geospatial SwissTNE dataset.

**🧰 Scripts (already exist, may need path updates)**

* extract\_nodes\_and\_edges.py
* write\_sumo\_nodes.py
* write\_sumo\_edges.py
* write\_empty\_connections.py
* generate\_net\_with\_netconvert.py
* summarize\_network\_contents.py

**📂 Key Inputs**

* swissTNE\_Base\_20240507.gpkg

**📄 Key Outputs**

* april\_2025\_swiss.net.xml (in SUMO/input/)

**🧩 Dependencies**

* None

**🔷 PHASE 2 – Vehicle Dataset Integration**

**📌 Goal**

Build SUMO-friendly vehicle specs from real train data.

**🧰 Scripts**

* merge\_vehicle\_data.py → builds merged\_jahresformation\_with\_vehicles.csv
* build\_vehicle\_profile\_table.py → builds vehicle\_profile\_table.csv
* [NEW] generate\_all\_vehicle\_type\_info.py → build vehicle\_type\_parameters.csv

**📂 Key Inputs**

* jahresformation.csv
* rollmaterial.csv
* rollmaterial-matching.csv

**📄 Key Outputs**

* vehicle\_type\_parameters.csv (SUMO-friendly vehicle attributes with bounds for randomization)

**🔷 PHASE 3 – Route Dataset Construction**

**📌 Goal**

Extract all realistic routes in Swiss data + match them to vehicle types and GTFS trip IDs

**📂 Output Files to Create**

* route\_vehicle\_info.csv  
  → [origin, destination, vehicle type(s), trip\_id, line, operator, etc.]

**🧰 New Script**

* [NEW] generate\_route\_vehicle\_info.py

**📂 Inputs**

* jahresformation.csv
* merged\_jahresformation\_with\_vehicles.csv
* GTFS files: trips.txt, routes.txt

**🔷 PHASE 4 – Route → Edge Mapping**

**📌 Goal**

Map each GTFS trip (origin → destination → intermediate stops) to the actual **edge list** in SUMO.

**📂 Output File**

* route\_edge\_mapping.csv  
  → [origin, destination, intermediate stops, SUMO edge sequence]

**🧰 New Script**

* [NEW] generate\_route\_edge\_mapping.py  
  → will use KDTree nearest-neighbor matching to map GTFS stops to SUMO nodes  
  → will find shortest path between stops using networkx

**📂 Inputs**

* GTFS: stop\_times.txt, stops.txt, trips.txt
* SUMO: rail\_nodes\_named.csv, rail\_edges\_named.csv, april\_2025\_swiss.net.xml

**🔷 PHASE 5 – Route Generator (Main Script)**

**📌 Goal**

Main control script to generate .rou.xml files dynamically.

**📁 Output**

* SUMO .rou.xml files for selected routes

**🧰 New Script**

* [NEW] route\_creator.py

**🔁 Logic**

1. Take input: origin + destination (+ optional vehicle type)
2. Use route\_edge\_mapping.csv to get edges
3. If vehicle type missing, infer from route\_vehicle\_info.csv
4. Get full parameters from vehicle\_type\_parameters.csv
5. Randomize within defined bounds (accel, decel, dwell time, sigma, etc.)
6. Write valid .rou.xml

**🔷 PHASE 6 – Baseline Scenario Simulations (non-VC)**

**📌 Goal**

Run train traffic simulations without virtual coupling.

**🧰 Script**

* [NEW] simulate\_scenario.py  
  → Uses TraCI to control batch runs

**📄 Output**

* tripinfo.xml, emissions.xml, logs per run

**🔷 PHASE 7 – KPI Logging & Visualization**

**📌 Goal**

Systematically extract performance indicators from .rou.xml runs.

**📁 Outputs**

* kpi\_summary.csv
* KPI plots (delay, headway, throughput, emissions, energy)

**🧰 New Scripts**

* extract\_kpis.py
* visualize\_kpis.py

**🔷 PHASE 8 – VC Module Preparation (Structure Only)**

**📌 Goal**

Create structure & stubs for:

* train\_manager.py
* vc\_coupling\_logic.py
* vc\_state\_tracker.py

VC logic will not be active in Progress Report 1 — just structured.

**🧭 TIMELINE & CHECKPOINTS**

| **Week** | **Task** |
| --- | --- |
| W1–2 | Rebuild SUMO network |
| W3–4 | Merge vehicle data + build profile table |
| W5–6 | Extract routes and edge mappings |
| W7–8 | Build dynamic route creator + run baseline simulations |
| W9–10 | Log KPIs + visualize results |
| W11 | Setup VC module stubs |
| W12 | Write Progress Report #1 content |

**🔐 Final Note**

Your code will:

* Run **only on your machine**
* Use **only local files**
* Be **100% modular, readable, and realistic**
* Serve both academic and industry goals

Would you like this step-by-step plan saved as a .md or .txt file for reference?