AI-Based Network Stabilization Tool: Project Plan

Document Version: 1.0 **Date:** July 31, 2025

1. Project Vision & Core Objective

This document outlines the plan of action for developing an AI-driven network stabilization tool. The core objective is to create an intelligent system that transforms network management from a reactive to a proactive, self-optimizing paradigm. The tool will be designed to manage modern hybrid networks, combining traditional wired infrastructure with LEO satellite communication links like Starlink.

The system will operate on a continuous "Analyze -> Predict -> Stabilize" loop, using a dual-AI strategy to:

1. **Proactively manage** predictable network events (e.g., satellite handoffs, traffic congestion) by intelligently steering traffic.
2. **Instantly detect and contain** major, unforeseen failures (e.g., configuration errors, software bugs) to prevent catastrophic outages.

2. Minimum Viable Product (MVP) Definition

The initial goal is to produce an MVP that validates the core concept.

- **Functionality**: The MVP will be a proof-of-concept application that monitors a simulated hybrid network (one wired path, one satellite path). It will use two parallel AI models to:
    1. **Predict** routine degradation on either path and automatically execute a **simulated traffic switch** to the healthier link.
    2. **Detect** a major, unforeseen anomaly on either path and trigger a **critical alert**, simulating a quarantine of the faulty link.
- **Goal**: To demonstrate, in a controlled environment, the system's ability to both proactively optimize and reactively contain failures, proving the value of the dual-AI approach.

3. High-Level Project Phases

The project is structured into four distinct phases, moving from initial simulation to a production-ready, real-world system.

Phase	Title	Time Required	Primary Goal

Phase 1	**Foundational Analysis & MVP Development**	**8 Weeks**	Develop a working MVP with a hybrid network simulator and a dual-AI core that can predict issues and demonstrate automated solutions.
Phase 2	**Advanced Modeling & Simulated Stabilization**	**12 Weeks**	Enhance AI models (e.g., with Transformers), refine the "playbook" of automated solutions, and begin research into Reinforcement Learning for true autonomous control.
Phase 3	**Real-World Integration & Edge Deployment**	**16 Weeks**	Integrate the system with live network hardware (wired routers, Starlink terminals), deploy inference models to the edge, and build out the production-grade cloud architecture.
Phase 4	**Full-Scale Stabilization & Commercialization**	**Ongoing**	Implement real control actions on the live network, scale the system to manage a fleet of devices, and continuously refine the AI with real-world data.

4. Phase 1 in Detail: Foundational Analysis & MVP Development (8 Weeks)

This phase is a focused sprint to build the MVP. The objective is to create a functional prototype that proves the core concepts are viable.

Week 1-2: Advanced Hybrid Network Simulation

- **Objective**: To create a robust data simulator that models a hybrid network with both predictable events and unpredictable failures.
- **Primary Roles**: Data/Software Engineer, with guidance from the AI/ML Scientist.
- **Key Tasks**:
    1. **Hybrid Schema Design**: Design a dataset with telemetry for two distinct paths: `wired` and `satellite`.
    2. **Predictable Event Injection**: Implement logic to inject routine, predictable events.
        - **Satellite Path**: Simulate periodic satellite handoffs by briefly spiking latency and jitter.
        - **Wired Path**: Simulate a predictable "network congestion" window based on time of day (e.g., peak business hours).
    3. **Unpredictable Failure Injection**: Implement logic to inject a random, major failure event. This simulates a "latent bug" or configuration error. For example, at a random point, the `wired_packet_loss` suddenly jumps to 80% and stays there.

4. **"Normal" Data Generation**: Ensure the simulator can generate long periods of clean, "normal" data for both paths, which is essential for training the anomaly detector.

- **Deliverable**: A Python script capable of generating a rich, hybrid dataset with labeled predictable events and unlabeled unpredictable failures.

Week 3-4: Unified Data Pipeline & Feature Engineering

- **Objective**: To build a data processing pipeline that prepares the hybrid data for both AI models.
- **Primary Roles**: AI/ML Scientist, Data/Software Engineer.
- **Key Tasks**:
    1. **Data Ingestion & Cleaning**: Load and clean the two-path dataset. Apply signal smoothing (e.g., Savitzky-Golay filter) to noisy fields like jitter.
    2. **Feature Creation**: Engineer features that help the models make decisions. A crucial new feature would be `current_optimal_path`, labeled based on which path has better metrics during predictable events.
    3. **Dataset Separation**: Create two distinct datasets from the same source:
        - **Dataset A (for Predictive Model)**: The full, labeled dataset containing all normal and predictable event data.
        - **Dataset B (for Anomaly Model)**: A carefully filtered dataset containing *only* the "normal" operational data.
- **Deliverable**: A data processing script that outputs two clean, feature-rich datasets ready for training two different AI models.

Week 5-6: Dual AI Model Training

- **Objective**: To train both the predictive steering model and the unsupervised anomaly detector.
- **Primary Roles**: AI/ML Scientist.
- **Key Tasks**:
    1. **Train Predictive Steering Model (Supervised)**:
        - Using **Dataset A**, train a supervised model (like an LSTM or simple Transformer) to predict the `current_optimal_path` for the next 60 seconds. It will learn the patterns that precede a handoff or congestion window.
    2. **Train Anomaly Detector (Unsupervised)**:
        - Using **Dataset B**, train an LSTM Autoencoder. Because it only ever sees "normal" data, it becomes an expert at identifying what is normal for both the wired and satellite links.
    3. **Model Validation**:
        - Validate the predictive model's accuracy in choosing the correct path.
        - Validate the anomaly detector by feeding it the unpredictable failure data and ensuring it produces a high "reconstruction error" score.
- **Deliverable**: Two trained and validated models: one for proactive path selection and one for critical failure detection.

Week 7-8: MVP Integration, Solution Logic, and Demo

- **Objective**: To build a functional MVP application that integrates both models and demonstrates automated solutions.
- **Primary Roles**: Entire Team.
- **Key Tasks**:
    1. **Application & GUI Development**: A Data/Software Engineer builds a Streamlit application that visualizes the real-time performance of both network paths.
    2. **Dual Model Integration**: The application loads both AI models and feeds them live simulated data in parallel.
    3. **Implement Solution Logic (Remediation Engine)**:
        - **Proactive Switch**: If the **predictive model** recommends switching from Path A to Path B, the GUI will show the active path changing and display a message: "**Proactive Switch**: Predicted congestion on Wired path. Rerouting to Satellite."
        - **Critical Failure Alert**: If the **anomaly detector** outputs a high error score for Path A, the GUI will show an immediate, high-priority alert and simulate taking that path offline: "**CRITICAL ANOMALY**: Unforeseen failure detected on Wired path. Path quarantined. Forcing traffic to Satellite."
    4. **Final Testing & Demo Prep**: The team conducts end-to-end testing and prepares a demonstration that showcases both a proactive switch during a routine event and a critical alert during a major failure.
- **Deliverable**: A working MVP application demonstrating a dual-strategy AI co-pilot for a hybrid network. This successfully concludes Phase 1.