**Business Requirements Specification**

**Project Title:** DTN ADMLOD Export Prototype  
**Prepared by:** Jordan Eisenback  
**Date:** October 5, 2025

**1. Project Objective**

Deliver a modular, testable C# command-line prototype that simulates integration with DTN’s ADMLOD interface. The prototype will:

* Generate ADMLOD-compliant export files using data from a mocked persistence layer.
* Simulate FTP upload behavior.
* Dynamically generate DTN-like response feedback based on export command validation.
* Use dependency injection to decouple CLI orchestration from business logic and data access.

**2. Scope**

**In Scope**

* CLI interface for triggering export operations.
* Repository-based mock data for Products and Product Groups.
* Dynamic generation of ADMLOD export files.
* Simulated FTP upload (local file write).
* Dynamic response generation based on export command validation.
* Structured logging with severity levels and correlation IDs.
* Dependency injection for all core services (data access, response generation, logging).
* Simulation of error scenarios:
  + Missing required fields
  + FTP failure
  + Invalid command format

**Out of Scope**

* Real FTP connectivity or live DTN integration.
* UI or web interface.
* Use of real credentials or sensitive data.
* Production deployment.

**3. Business Rules**

* All data is loaded via repository interfaces (e.g., JSON-backed or in-memory).
* Export logic must follow ADMLOD format and command structure.
* Response feedback is generated dynamically based on command validity.
* All services must be injected via interfaces to support modularity and testing.
* Logging must include timestamps, severity, and correlation IDs.
* Error handling must categorize failures (validation, FTP, parsing) and support retry logic where applicable.

**4. Success Criteria**

* Prototype generates ADMLOD export files from repository-backed mock data.
* FTP upload is simulated and logged.
* Response feedback is dynamically generated and reflects command validity.
* CLI supports flexible testing via flags (--success, --error, --logfile).
* All services are injected and testable in isolation.
* Stakeholders can review logs and outputs for traceability and auditability.

**5. Assumptions**

* ADMLOD format and command structure are based on DTN documentation.
* No real credentials or sensitive data are used.
* All mock data and configuration files are provided or created as part of the project.
* Prototype is for demonstration and planning only.

**6. Constraints**

* Must be implemented in C# (.NET 6 or later).
* Must run as a cross-platform command-line application.
* All integrations (FTP, DTN) are mocked.
* Configuration must be externalized (e.g., appsettings.json).
* All core logic must be modular and injected via interfaces.