**Project Setup**

* Initialize a new C# .NET console application repository.
* Set up the directory structure, including /MockFiles for mock data and responses.
* Add a README with build/run instructions.

**Mock Data Implementation**

* Create MockData.cs to provide hardcoded or file-based mock Products and Product Groups.
* Include at least two valid products and two valid product groups.
* Add at least one product/group with a missing required field for error simulation.

**ADMLOD Export File Generation**

* Implement AdmLodFileGen.cs to generate ADMLOD export files from mock data.
* Ensure the file includes the correct header, export command, and one line per product/group.
* Save the generated file to /MockFiles/adm\_export.txt.

**FTP Upload Simulation**

* Implement FtpClientMock.cs to simulate FTP upload by writing the export file to /MockFiles.
* Log the upload action.
* Add logic to simulate FTP failure (e.g., via a CLI flag or test case).

**Mock DTN Response Files**

* Create adm\_response\_success.txt and adm\_response\_error.txt in /MockFiles.
* Ensure the success file contains only successful responses.
* Ensure the error file contains at least one error (e.g., missing field, invalid command).

**Response Parsing**

* Implement ResponseParser.cs to read and parse the selected mock DTN response file.
* Output parsed results to the console, clearly distinguishing between successes and errors.

**Logging**

* Implement Logger.cs to log all actions and errors to the console.
* Add support for logging to a file if a --logfile argument is provided.

**CLI Argument Handling**

* Implement CLI parsing in Program.cs to support:
  + export command
  + --success or --error flag for response file selection
  + --logfile <filename> for optional file logging
* Validate and handle invalid or missing arguments.

**Error Scenario Simulation**

* Add logic to simulate and log:
  + Missing required fields in mock data
  + FTP upload failure
  + Invalid command format in export file
  + Errors in DTN response file

**Output Review and Acceptance**

* Ensure the generated export file matches ADMLOD format.
* Ensure parsed response output is clear and matches the mock response file.
* Demonstrate both successful and error scenarios.

**Documentation**

* Update the README with:
  + Build and run instructions
  + CLI usage examples
  + Description of mock data and error simulation
* Document the structure of mock files and how to modify them for testing.

**Testing**

* Execute all test cases from the Test Plan.
* Log results and defects.
* Verify acceptance criteria for each user story.