1. As a Dataload Service User, I should have a WebAPI that would accept valid (XML or JSON format) Dataload requests and on success should return “request accepted (202) “ otherwise return error with appropriate error message
2. As a Dataload Service User, I should have a WebAPI that authorize the fourth users requesting the Dataload from a central location else would return an unauthorized access error
3. As a Dataload Service User, I should have a restful API endpoint that I can asynchronously upload the data from into portal database so that I do not directly lock any of the DBOs and follow Fourth standards of using Azure bus, staging DB, using a windows service in the process.
   1. Take the valid Dataload request and run server validations on the records, validation logic will reside in a separate assembly.
   2. Build a stream of executable commands with parameters from data that passes the validation.
   3. Connect to staging DB through staging repository and upload the Dataload parameters with their sequence of execution request and unique Dataload Request ID.
   4. Connect to Orchestration message bus and upload the command with Dataload Request ID into the queue for the data that it has uploaded.
   5. Read message bus queue and the staging DB for any Dataload requests and rebuild the commands collection with parameters.
   6. Process the commands built out of bus and staging DB
   7. Bulk upload inside PS DB after processing the data with its result.
   8. Define canonical models inside orchestration to integrate between disparate parts of the application.
4. As a Dataload Service User, I should have a mechanism to purge the data that gets accumulated inside staging DB on regular basis so that staging DB remains within a manageable size and respond to service request promptly.
5. As a Dataload Service User, I should be able to view the list of Dataloads I requested in the past by Dataload types so that I can pick them and audit
6. As a Dataload Service User, I should be able to select and audit the Dataloads that I requested to be run in the past so that I can investigate the success, failure, and reason for a failure on each of the dataload requested